

Development and feasibility of economical hardware and software in control
theory application

by

Derek J. Black

B.S., Kansas State University, 2014

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Mechanical and Nuclear Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2017

Approved by:

Major Professor
Dr. Dale Schinstock

Copyright

DEREK J. BLACK

2017

Abstract

Control theory is the study of feedback systems, and a methodology investigated by many engineering students throughout most universities. Because of control theory's broad and interdisciplinary nature, it necessitates further study by application through experimental learning and laboratory practice. Typically, the hardware used to connect the theoretical aspects of controls to the practical can be expensive, big, and time consuming to the students and instructors teaching on the equipment. Alternatively, using cheaper sensors and hardware, such as encoders and motor drivers, can obfuscate the collected data in a way that creates a disconnect between developed theoretical models and actual system results. This disconnect can dissuade the idea that systems can and will follow a modeled behavior.

This thesis attempts to assess the feasibility of a piece of laboratory apparatus named the NERMLAB. Multiple experiments will be conducted on the NERMLAB system and compared against time-tested hardware to demonstrate the practicality of the NERMLAB system in control theory application.

Table of Contents

| | |
|--------------------------------------------------|------|
| List of Figures | viii |
| List of Tables | xi |
| Acronyms | xiii |
| 1 Introduction | 1 |
| 2 Apparatus | 3 |
| 2.1 NERMLAB | 3 |
| 2.1.1 NERMLAB Hardware | 3 |
| 2.1.2 Position Sensor | 5 |
| 2.1.3 NERMLAB Parts | 9 |
| 2.1.4 NERMLAB Cost | 10 |
| 2.1.5 NERMLAB GUI | 11 |
| 2.2 Motorlab | 12 |
| 2.2.1 Motorlab Hardware | 13 |
| 2.2.2 Motorlab GUI | 13 |
| 3 System Identification | 15 |
| 3.1 Motor Resistance | 16 |
| 3.1.1 Resistance Estimation | 16 |
| 3.2 Motor Torque Constant and Back EMF | 17 |
| 3.2.1 Estimating the Back EMF Constant | 18 |
| 3.3 Mass Moment of Inertia Estimation | 21 |

| | | |
|-------|-----------------------------------------------------------------------|----|
| 3.3.1 | Software Modeling of Mass Moment of Inertia | 22 |
| 3.3.2 | Mathematical Approximation of Mass Moment of Inertia | 22 |
| 3.3.3 | Lumped Mass Moment of Inertia | 24 |
| 3.4 | Motor Inductance | 24 |
| 3.4.1 | Inductance Estimate | 25 |
| 3.5 | Viscous Friction | 26 |
| 4 | Development of Mathematical Models for NERMLAB | 27 |
| 4.1 | Brushless Motor Theory | 27 |
| 4.2 | Electrical Dynamics | 29 |
| 4.3 | Combined Dynamics - Electrical and Mechanical | 30 |
| 4.3.1 | Position Models | 31 |
| 4.3.2 | Speed Models | 33 |
| 4.3.3 | Low-Pass Filter | 34 |
| 5 | Approximating Friction of a BLDC Motor | 35 |
| 5.1 | Friction Estimate | 36 |
| 5.2 | Speed Decay | 38 |
| 5.3 | Motorlab Results | 39 |
| 6 | Frequency of Oscillation of a Position Control System | 41 |
| 6.1 | Mathematical Model of a Closed-Loop Position Control System | 42 |
| 6.2 | Experiment | 43 |
| 6.3 | Motorlab Results | 47 |
| 7 | High Frequency Dynamics | 50 |
| 7.1 | Mathematical Models | 51 |
| 7.2 | Experiment | 52 |
| 7.3 | Motorlab Results | 56 |

| | | |
|------|-------------------------------------------------------------|-----|
| 8 | Frequency Response of a Position Control System | 58 |
| 8.1 | Mathematical Model | 59 |
| 8.2 | Experiment | 59 |
| 8.3 | Motorlab Results | 64 |
| 9 | Effects of Integral Control on Steady State Error | 67 |
| 9.1 | Mathematical Model | 68 |
| 9.2 | Experiment | 69 |
| 9.3 | Motorlab Results | 73 |
| 10 | Derivative Action on a Position Control System | 76 |
| 10.1 | Mathematical Model Development | 77 |
| 10.2 | Experiment | 78 |
| 10.3 | Motorlab Results | 83 |
| 11 | Conclusion | 85 |
| | Bibliography | 87 |
| A | Motorlab Specifications | 89 |
| B | Part Drawings | 109 |
| C | Laboratory Procedures | 113 |
| D | Code | 134 |
| E | System Specifications | 143 |
| F | Experiment Documentation | 145 |
| G | Additional Models | 147 |
| G.1 | Pendulum Model | 147 |

| | | |
|-------|------------------------------|-----|
| G.2 | State Space Models | 149 |
| G.2.1 | Position Models | 150 |
| G.2.2 | Speed Models | 153 |

List of Figures

| | | |
|-----|-----------------------------------------------------------|----|
| 2.1 | NERMLAB | 4 |
| 2.2 | Magnet and AS5047D | 6 |
| 2.3 | Sensor Noise with Changing Speed | 7 |
| 2.4 | Section View of Motorlab Assembly | 9 |
| 2.5 | NERMLAB GUI | 11 |
| 2.6 | Motorlab | 12 |
| 2.7 | Motorlab GUI in MATLAB | 13 |
| 3.1 | Motor Connection Configuration | 16 |
| 3.2 | Measured Back EMF vs Speed | 20 |
| 3.3 | Inductance Measurement Circuit | 25 |
| 4.1 | Electrical and Mechanical Schematic of NERMLAB | 29 |
| 4.2 | Position Model | 31 |
| 4.3 | NERMLAB Speed Model | 33 |
| 4.4 | Low-Pass Filter Bode | 34 |
| 5.1 | Viscous Friction Estimate | 37 |
| 5.2 | NERMLAB Speed Decay | 39 |
| 5.3 | Motorlab Speed Decay | 40 |
| 6.1 | Block Diagram of Closed Loop Control System | 42 |
| 6.2 | Block Diagram Reduction | 43 |
| 6.3 | Root Locus of a Position Control System | 44 |
| 6.4 | NERMLAB Step Responses for a Set of Three Gains | 46 |

| | | |
|------|---------------------------------------------------------------------------|-----|
| 6.5 | NERMLAB Step Responses for a Set Outside Experimental Range | 47 |
| 6.6 | Motorlab Step Responses for a Set of Three Gains | 48 |
| 7.1 | Block Diagram of Closed Loop Speed Control System | 51 |
| 7.2 | Block Diagram of Open Loop System (Speed Control) | 51 |
| 7.3 | NERMLAB Step Responses for a Set of Three Gains | 54 |
| 7.4 | NERMLAB Step Responses for Unstable Proportional Gain | 55 |
| 7.5 | Motorlab Step Responses for a Set of Two Gains | 56 |
| 8.1 | Bode Plot - Position Control System | 60 |
| 8.2 | Phase Lag | 63 |
| 8.3 | 90 Degree Phase Shift | 63 |
| 8.4 | Bode Plot of a Second Order Spring System | 65 |
| 9.1 | Block Diagram of Closed Loop Speed Control System with PI Control | 68 |
| 9.2 | NERMLAB Steady State Error | 71 |
| 9.3 | Steady State Error at Higher Proportional Gain | 72 |
| 9.4 | Motorlab Steady State Error | 74 |
| 10.1 | Root Locus for PD Controller on NERMLAB | 77 |
| 10.2 | Block Diagram of Closed Loop Control System | 78 |
| 10.3 | NERMLAB PD Controller Experimental Results | 80 |
| 10.4 | Period of Oscillation Calculation | 81 |
| 10.5 | Angle Error of PD Control System | 82 |
| 10.6 | Steady State Error at Higher Proportional Gain | 82 |
| 10.7 | Motorlab PD Controller Experimental Results | 84 |
| B.1 | Standoff for NERMLAB | 110 |
| B.2 | Magnet holder for NERMLAB | 111 |
| B.3 | Torque Transmission Shaft for NERMLAB | 112 |

| | | |
|-----|----------------------------------------------------------------|-----|
| D.1 | Encoder Noise Experiment | 135 |
| D.2 | Back-EMF Plotter | 136 |
| D.3 | Friction Laboratory | 137 |
| D.4 | Root Locus Plotter | 138 |
| D.5 | Frequency of Oscillation Position Control Code | 139 |
| D.6 | Frequency Response of a Position Control System Code | 140 |
| D.7 | Steady State Error Code | 141 |
| D.8 | Proportional-Derivative Experiment Code | 142 |
| F.1 | Back emf at various motor speeds | 146 |
| G.1 | NERMLAB Pendulum Model | 148 |

List of Tables

| | | |
|-----|--------------------------------------------------------------------------|----|
| 2.1 | Noise Experiment | 8 |
| 2.2 | Motorlab expenditure report | 10 |
| 3.1 | Motor Parameters Nomenclature | 15 |
| 3.2 | Measured motor resistance | 17 |
| 3.3 | Measured back voltage | 19 |
| 3.4 | Measured Inertia Parameters | 21 |
| 3.5 | Inductance Experiment | 26 |
| 5.1 | Friction Experiment | 38 |
| 6.1 | Table of Experimental Gains | 43 |
| 6.2 | Experimental Results for NERMLAB | 45 |
| 6.3 | Experimental Results for Motorlab | 49 |
| 7.1 | Table of Experimental Gains | 53 |
| 7.2 | Table of Experimental Motorlab Gains | 57 |
| 8.1 | Experimental Results for Position Control Frequency Response | 60 |
| 8.2 | Experimental Results for Position Control Frequency Response on Motorlab | 66 |
| 9.1 | Steady State Error | 68 |
| 9.2 | Table of Experimental Gains | 70 |
| 9.3 | Experimental Results for NERMLAB | 72 |
| 9.4 | Steady State Error | 73 |
| 9.5 | Experimental Results for Motorlab | 75 |

| | | |
|------|------------------------------------------------------|-----|
| 10.1 | Gains for PD Controller | 78 |
| 10.2 | Motorlab Gains for PD Controller | 83 |
| E.1 | NERMLAB Parameters | 143 |
| E.2 | Motorlab Parameters | 144 |
| F.1 | Viscous Friction Full Experimental Results | 145 |

Acronyms

| | |
|-----------------|----------------------------------|
| ARM | Advanced RISC Machine |
| DAEC | Dynamic Angle Error Compensation |
| MPU | Microprocessor Unit |
| BLDC | Brushless DC |
| GUI | Graphical User Interface |
| NERMLAB | New Earth Robotics Motor Lab |
| back-emf | back electromotive force |
| RPM | Rotations Per Minute |
| CAD | Computer Aided Design |
| SPI | Serial Peripheral Interface Bus |
| PWM | Pulse Width Modulation |
| JSON | JavaScript Object Notation |
| P | Proportional |
| PI | Proportional-Integral |
| PD | Proportional-Derivative |
| PID | Proportional-Integral-Derivative |
| ZPK | Zero-Pole-Gain |

Chapter 1

Introduction

Current research indicates a growing need for laboratory components for introductory control theory classes. However, many hurdles like budget, class size, and space limitations arise when laboratories are appended to lectures in universities [1, 2, 12]. The New Earth Robotics Motor Lab (NERMLAB) aims to address these concerns in reducing the overall cost imposed on instructors and students, as well as, minimizing the foot print of the hardware to allow students to take part in laboratories in a home environment. It is this home experimentation that allows students to engage in experimental learning, which is a methodology that aims at creating knowledge through wisdom, observation, and insight from experience. Experimental learning also provides an alternative learning mechanism for the traditional theoretical components that make up a standard engineering curriculum. Most experimental learning is achieved through a laboratory practicum that helps students connect the theoretical ideas developed in lecture with what is done in practice [12]. As a result, students can gain further insight into the theory that might have gone unresolved without experimental learning [2].

Unfortunately, classroom sizes continue to grow in universities, and, a direct result of this, is increasing laboratory size. Since size means the cost per student increases due to the limited amount of equipment available, there is a desire for more affordable hardware [3]. A way to combat the issue would be to make laboratory hardware more portable, allowing for

cheaper components, such as motors, motor drivers and the like to be used [2]. However, utilizing cost-effective hardware in laboratory equipment can lead to poorly produced data, which does not adhere to theoretical models developed in lecture. While it is true that cheaper hardware does lead to less accurate data, it does allow greater access to students because of its cost. The goal of NERMLAB is to give students access to affordable equipment that can provide them with the experimental learning opportunities both in the classroom and at home. In addition, it is the aspect of portability that is of importance because students will be allowed to learn at their own pace, in a way that benefits them the most, and still achieve the same learning objective as that of traditional on-campus laboratories [2, 4]. Other studies, such as [12], also have compared more affordable laboratory apparatus to their more expensive counterparts to determine exactly how learning objectives affected. [12] concludes that even when cheaper and portable laboratory kits are used, the same learning objectives are achieved.

This thesis will attempt to address the feasibility of the cheaper NERMLAB alternative. Multiple experiments will be conducted as they appear in Appendix C and results will be compared to more expensive hardware, such as the Motorlab. Chapter 2 will describe the NERMLAB system apparatus and the various components that comprise it, as well as, comment on the differences between the NERMLAB and the older Motorlab system. Then, Chapter 3 will discuss system identification and characterization, which produces things such as the motor torque constant, inductance, and resistance. Chapter 4 will then develop the necessary mathematical models that are necessary for the experiments that make up Chapters 5-10.

Chapter 2

Apparatus

This chapter will discuss two apparatus pieces used for conducting a series of five experiments included in this thesis and will, in detail, describe the purpose, design, and creation of the equipment. Section 2.1 will describe the NERMLAB, including the hardware implementation, design of components, basic functionality, and use of the position sensor. In comparison, section 2.2 will describe the Motorlab, the model currently being used in Kansas State University’s engineering laboratories, and how it differs from NERMLAB.

2.1 NERMLAB

The NERMLAB is a reimplementation of older laboratory hardware created by Dr. Dale Schinstock and Dr. Warren White for Control of Mechanical Systems I at Kansas State University. This equipment allows users to connect the theoretical ideas of control theory with those in practice.

2.1.1 NERMLAB Hardware

The NERMLAB consists of several key pieces of hardware, including: an STM32 Nucleo development board, motor driver, and a Brushless DC (BLDC) motor (Figure 2.1). The

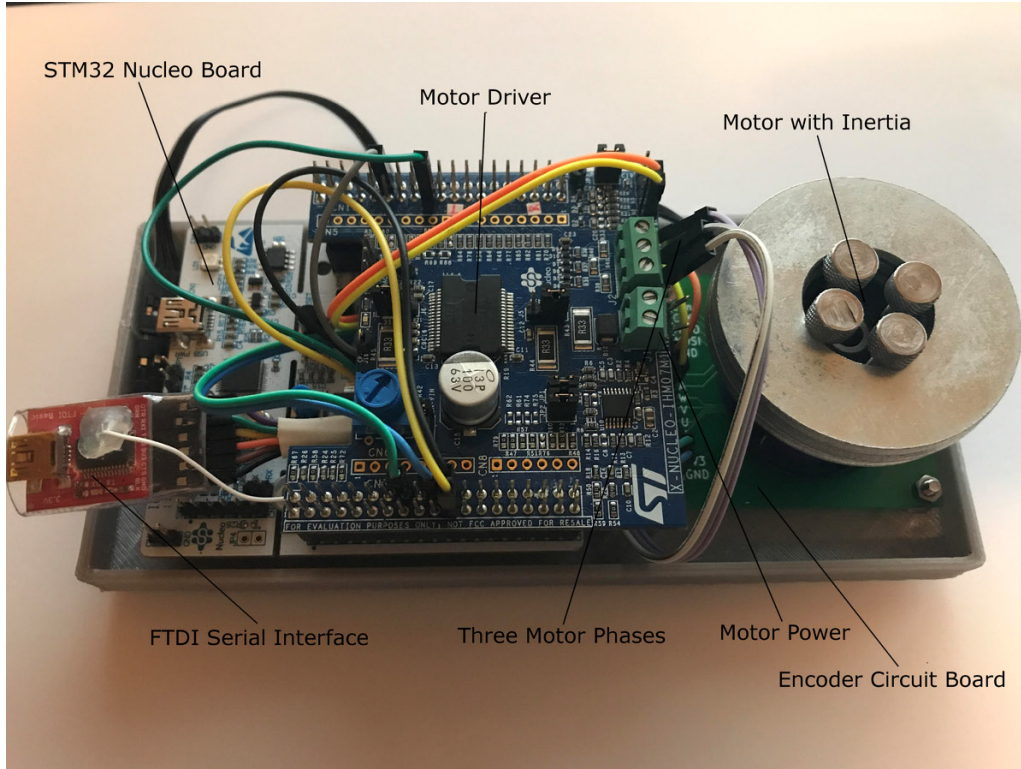


Figure 2.1: NERMLAB

STM32 Nucleo houses a STM32F401RE Microprocessor Unit (MPU), which is a 32-bit processor with an 84 MHz clock speed and up to 512 Kbytes of flash memory. The STM32 Nucleo also allows Arduino shields and other STM boards to be attached for added functionality.

A motor driver was required to drive a brushless DC motor. As a result, an X-Nucleo-IHM07M1 (a three-phase brushless DC motor driver) was selected to be the primary driver for the NERMLAB. The X-Nucleo has a nominal operating voltage of 8V-48 VDC with a 2.8 A peak current output, which is sufficient to drive a BLDC gimbal motor, such as the RCTIMER GBM2804, which is the primary motor used in this thesis.

The RCTIMER GBM2804 is a 100 turn BLDC motor that has a hollow shaft, which allows placement of a position sensor for feedback control purposes. Motor specifications were not given by the manufacturer of this motor, so Chapter 3 details the experiments that were conducted to find the various parameters needed to adequately model the entire NERMLAB system.

2.1.2 Position Sensor

The main purpose of the NERMLAB is to conduct control laboratory experiments. To accomplish this, feedback via sensor readings is necessary, and the typical way to do position and speed control is to use position feedback via an encoder. An encoder is a device that converts angular position of a motor shaft to an analog or digital signal that can be processed by an MPU. In the case of the NERMLAB, an on-axis magnetic encoder is used to do position feedback. Special equipment had to be designed in order to use this type of encoder, and will be detailed in Section 2.1.3.

The encoder that is being used consists of 14-bit on-axis magnetic rotary position sensor chip, specifically the AS5047D by AMS ¹. The position sensor chip provides high resolution absolute angle measurements through a full 360 degree range ². In addition to the fast absolute angle measurement system that the position sensor provides, it also has Dynamic Angle Error Compensation (DAEC) that provides position control systems with near 0 latency [5].

The AS5047D chip is a magnetic sensor that utilizes the Hall-effect. The chip works by taking the Hall sensors and converting the perpendicular magnetic field on the surface of the chip to a voltage. The voltage signals are filtered and amplified in order to calculate the angle of the magnetic vector. In order for position measurements to be taken, a small diametrically opposed magnet must be placed on the shaft of the equipment being measured. The magnet and AS5047D are contactless, meaning there is a small air gap between the chip and magnet. As the magnet rotates above the chip (Figure 2.2), angle measurements are calculated and transmitted through the chip [5].

Sensor Output

The AS5047D has multiple input/output types that can be used for feedback and chip programming. A Serial Peripheral Interface Bus (SPI) is the main input to the chip that

¹AMS is an Austrian analog sensor and semi-conductor manufacturer

²These chips typically provide a maximum resolution of 2000 steps/revolution in decimal mode and 2048 steps/revolution in binary mode

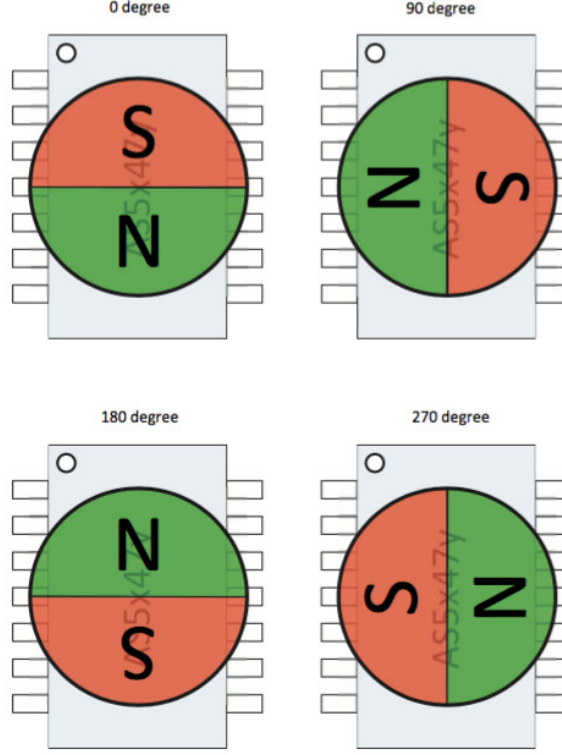


Figure 2.2: Magnet and AS5047D [5]

allows a one time programming operation to be carried out. The chip also outputs an ABI and Pulse Width Modulation (PWM) signal that can be used in feedback measurements. In the case of the NERMLAB, the ABI output is the chosen signal type to be used for encoder readings. The ABI is an incremental type signal that has two 90 degree offset signals that indicate motor direction. To determine the motor's position, one only needs to count the number of pulses coming from the chip either from the leading or falling edge of the signal. From there it is possible to use Equation 2.1 to come up with the position in radians, where n_{resol} is the resolution of the encoder output and n_{count} is the current encoder count.

$$\theta_{rad} = 2\pi \frac{n_{count}}{n_{resol}} \quad (2.1)$$

PWM is also used on the NERMLAB, which provides absolute angle measurements through a full 360 degree range. One PWM clock period on the AS5047D represents 0.088

degrees, with a 12-bit resolution. However, for the purposes of control, the PWM signal is not adequate, as the update rate is less than ten times the bandwidth of the system. Rather, this signal is used to aid the motor commutation of the BLDC motor.

Sensor Noise

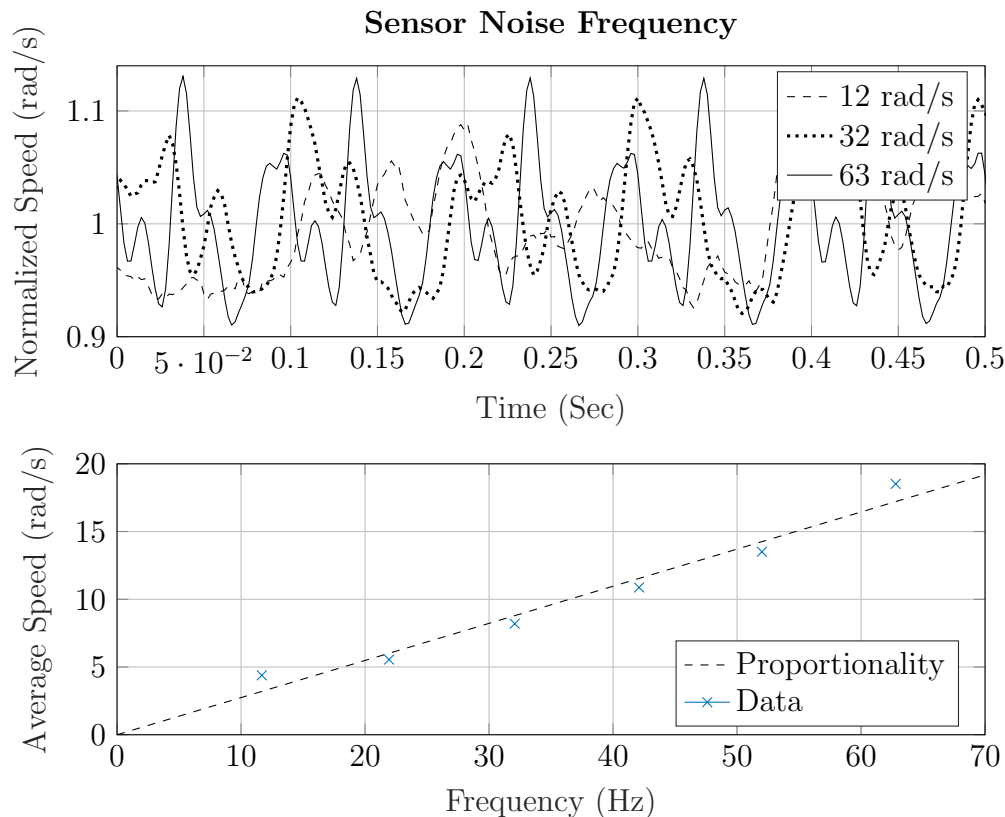


Figure 2.3: Sensor Noise with Changing Speed

While the AS5047D does have a high resolution output in comparison to other cheap encoders, the chip does suffer from measurement inaccuracies. These inaccuracies could be a result of the magnet's rotation or the tolerance and fit of the magnet holder. It is when doing speed control that these inaccuracies show up on the NERMLAB as superimposed noise on the response. It was found through experimentation that the frequency of the noise was proportional to the speed of the rotor. The NERMLAB was run at incremental speeds, and the frequency of the noise was tabulated in Table 2.1. It is evident from Figure 2.3 that

as the speed of the rotor is increased, the frequency of the noise increases proportionally, as described by Equation 2.2.

$$f = 0.2741v \quad (2.2)$$

Table 2.1: Noise Experiment

| Average Speed (rad/s) | Frequency (Hz) | Input Voltage (V) |
|-----------------------|----------------|-------------------|
| 11.67 | 4.386 | 1.5 |
| 21.93 | 5.55 | 2.5 |
| 32.07 | 8.196 | 3.5 |
| 42.10 | 10.869 | 4.5 |
| 52.00 | 13.510 | 5.5 |
| 62.79 | 18.518 | 6.5 |

While the exact problem is not pinpointed in this thesis, the measurement inaccuracy does not greatly affect the experimental results and is more so another feature that must be realized when doing system analysis.

2.1.3 NERMLAB Parts

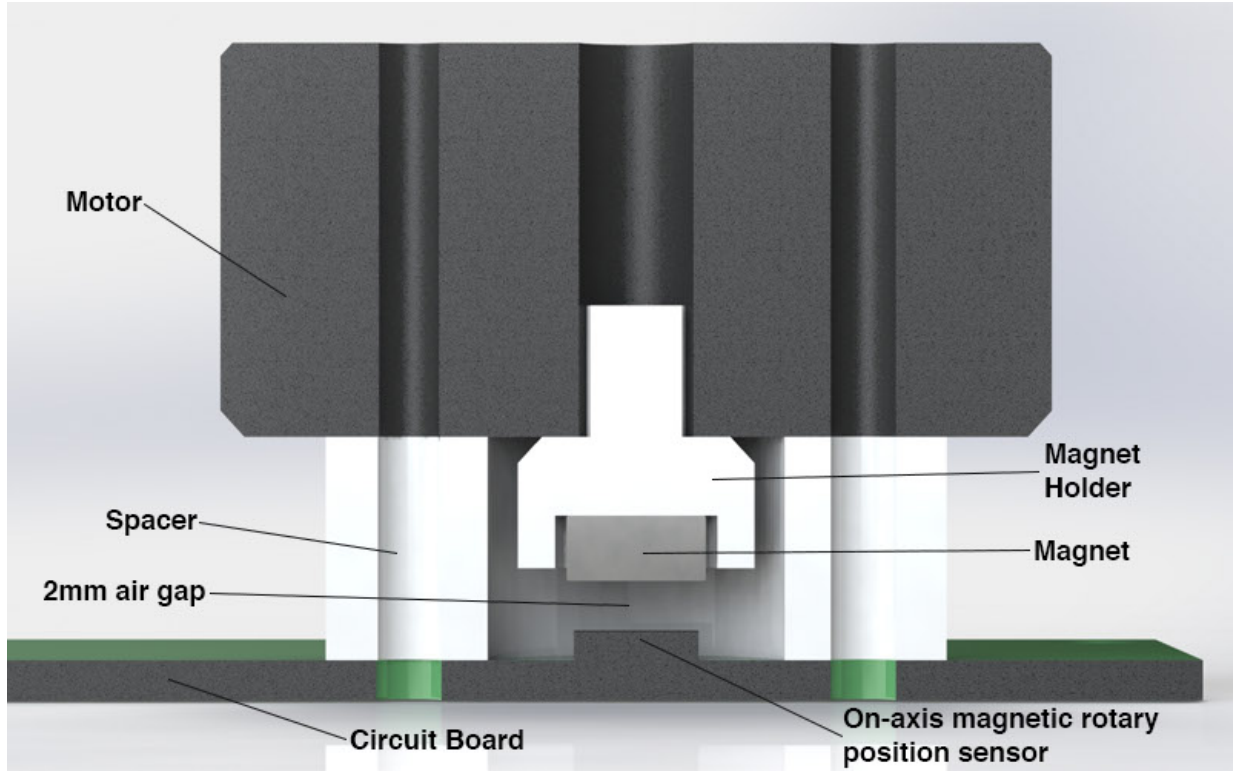


Figure 2.4: Section View of Motorlab Assembly

Along with the hardware mentioned in Section 2.1.1, three components were needed to be developed in order to bring the NERMLAB to fruition: a printed circuit board that houses the on-axis magnetic rotary position sensor, a spacer to put distance between the circuit board and the motor, and a magnet holder, which holds one diametrically opposed magnet ³. Both the spacer and magnet holder, which can be seen in Figure 2.4, had to be 3D printed in order to achieve the required specifications of the apparatus setup. Detailed drawings of these two parts can be found in Appendix B, Figures B.1 and B.2, if reproduction is desired. Along with the two 3D printed parts, a printed circuit board had to be designed to allow the position sensor to communicate with the rest of the hardware.

Because of variability in resolution of current 3D printers, care was given to the design

³Diametrically opposed meaning the north and south poles of the magnet are in-plane as opposed to top/bottom poles. Reference Figure 2.2 for further clarification

of the magnet holder ⁴. A spline was used for both the shaft of the magnet holder and the section that holds the magnet itself. The spline allowed for greater tolerances in the parts, meaning the magnet holder could be easier to press fit into the motor, and likewise, allowed easier removal of the diametrically opposed magnet.

2.1.4 NERMLAB Cost

Table 2.2 lists the components that make up the NERMLAB system and their associated price at the writing of this thesis.

Table 2.2: Motorlab expenditure report

| Component | Brand/Manufacture | Cost |
|-----------------------|--------------------|------------------|
| BLDC Motor | RCTIMER GBM2804 | 11.94 USD |
| Position Sensor | AS5047D AMS | 4.21 USD |
| ST32 Nucleo | STMicroelectronics | 10.12 USD |
| X-Nucleo-IHM07M1 | STMicroelectronics | 9.80 USD |
| Magnet | - | 3.00 USD |
| Printed Circuit Board | - | 30.00 USD |
| TOTAL COST | | 69.07 USD |

⁴Because of this variability in resolution, the magnet holder was printed in iterations, varying the diameter.

2.1.5 NERMLAB GUI

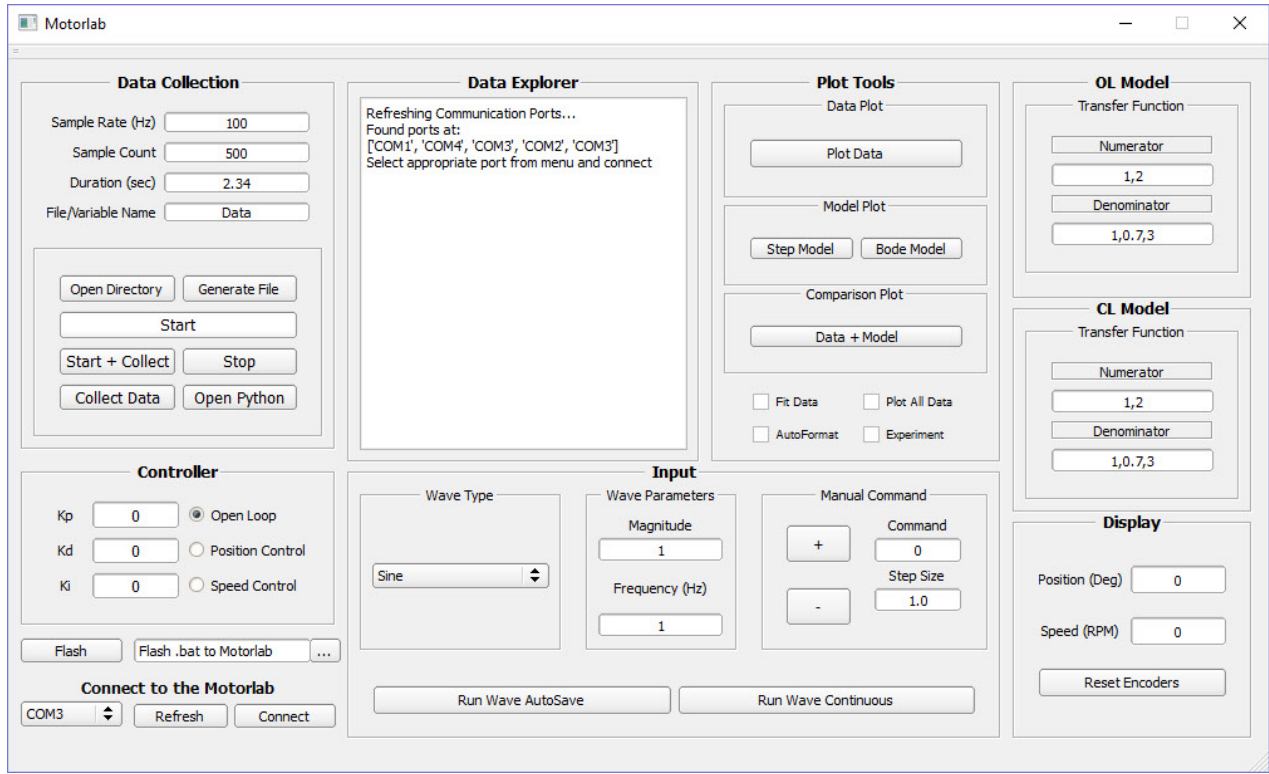


Figure 2.5: NERMLAB GUI

The NERMLAB GUI is an alternative way of interacting with the NERMLAB without having to hardcode values and re-flashing every time an experiment needs to be run. The user simply enters the desired settings into the GUI, which then triggers serial events in the back-end. The GUI is coded in python 2.7, utilizing a variety of different libraries, such as the PyQT4 framework that allows cross platform development of GUI applications, matplotlib for plotting purposes, and variety of signal processing toolboxes for employing frequency response and model responses.

JavaScript Object Notation (JSON) is the main communication protocol that is used to allow back and forth communication between the NERMLAB and GUI interface. The GUI sends out JSON messages whenever a user triggers an event, and the back-end code of the NERMLAB then sees the object in its buffer, which is decoded into key-value pairings that

can be processed.

Much of the NERMLAB GUI is still in development and is left as future work. Features that still need implementation would include: flashing hex and binary files directly to the NERMLAB system from the GUI, allowing python code to interact with the GUI/NERMLAB system, and improving the plotting capabilities for instant data visualization.

2.2 Motorlab

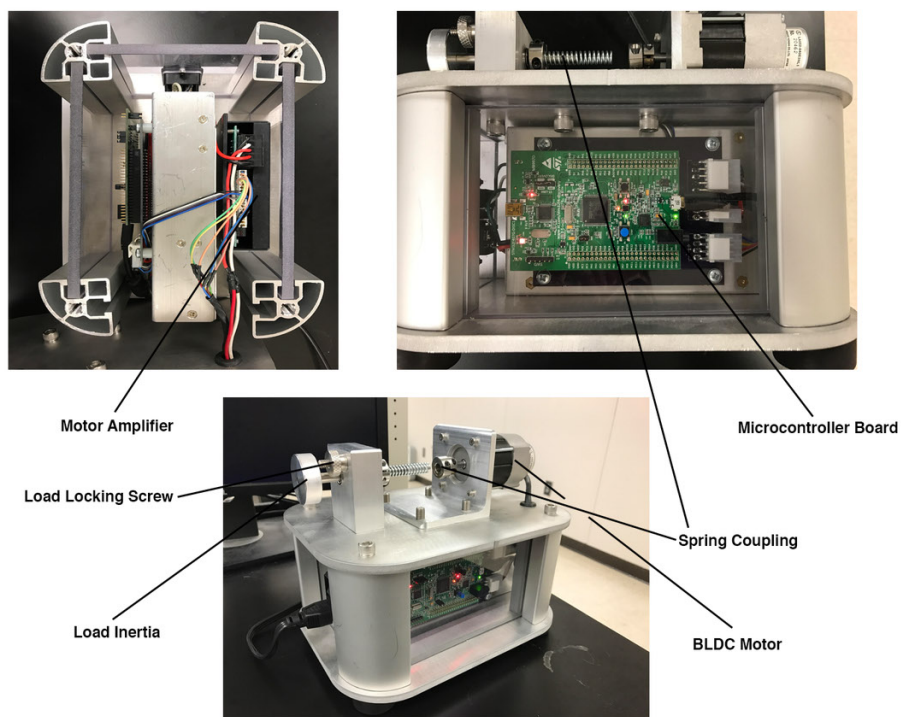


Figure 2.6: Motorlab

The Motorlab has been in service at Kansas State University for over 15 years, and as a result, is a time-tested piece of laboratory hardware that has proven to be reliable in providing quality data. Additionally, the Motorlab's hardware components were selected to ensure a very clear translation of laboratory results, allowing the Motorlab to have a much larger operating limit and bandwidth than students use in laboratory. As a result, the

Motorlab represents a good base model to which to compare the results of the NERMLAB apparatus in this thesis to.

2.2.1 Motorlab Hardware

Various hardware make up the Motorlab, namely, a BLDC motor, BLDC servo amplifier by Copley Controls Corp., and a ST Discovery board ⁵. Typically the Motorlabs run a cost of about 700 USD per lab station [3]; however, this estimate does not include things like manufacturing, design and labor, as these processes were carried out at Kansas State University. Appendix E, Figure E.2 hosts the system parameters that make up the Motorlab.

2.2.2 Motorlab GUI

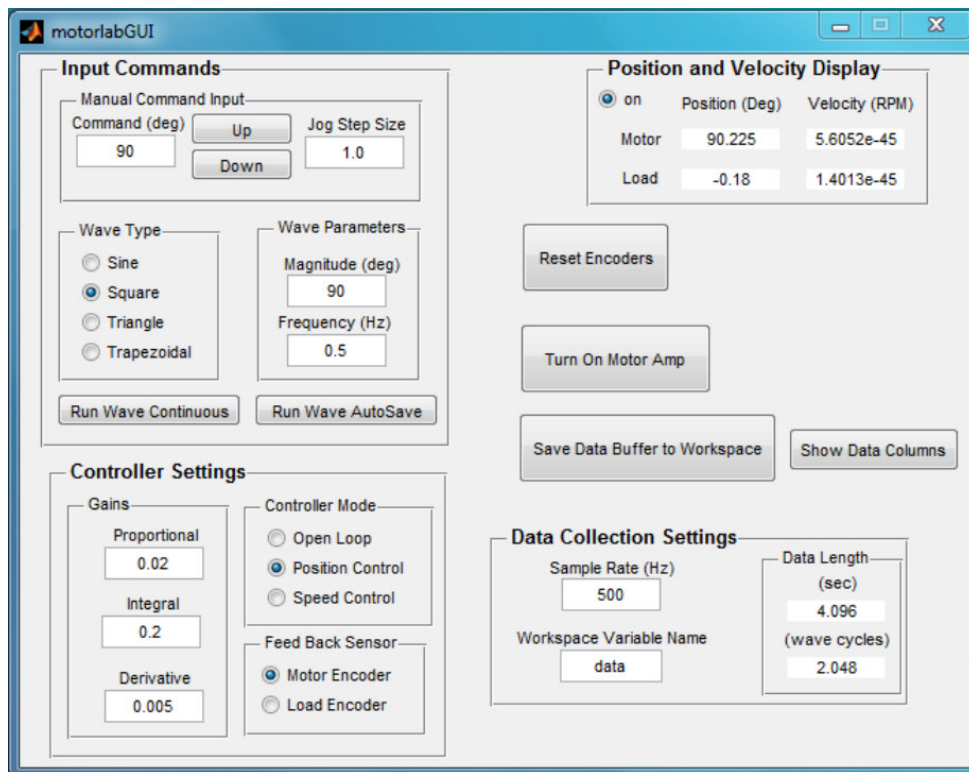


Figure 2.7: Motorlab GUI in MATLAB

⁵The ST Discovery Board is a 32-bit micro controller development board

The Motorlab interfaces with a Graphical User Interface (GUI) coded in MATLAB to allow users to run various laboratory experiments on the hardware. It allows the selection of various wave types, frequency, controller gains, and sample rate that get sent to the Motorlab. After the parameters of the experiment are setup, the GUI can run the Motorlab, which in turn sends the experimental data to the workspace of MATLAB in the form of a matrix. The MATLAB GUI can be seen in Figure 2.7.

Chapter 3

System Identification

Chapter 3 will be dedicated to developing the various parameters that make up the NERMLAB, such as the motor torque constant, back electromotive force (back-emf), inductance, and max voltage. Each section in Chapter 3 will detail the process of how the various parameters were measured, calculated, and experimentally determined. Nomenclature for various constants and parameters are detailed in Table 3.1.

Table 3.1: Motor Parameters Nomenclature

| Parameter | Description |
|-----------|-----------------------------------------------------------|
| V | Motor Voltage |
| k_t | Motor Torque Constant in dq Reference Frame |
| k_T | Overall Motor Torque Constant |
| K_e | Line-Line Back Electromotive Force Constant |
| K_E | Back Electromotive Force Constant in dq Reference Frame |
| $K_{e,p}$ | Back Electromotive Force Constant per Phase |
| J | Lumped Mass Moment of Inertia ($3J_w + J_r + 4J_b$) |
| J_w | Washer Mass Moment of Inertia |
| J_r | Rotor Mass Moment of Inertia |
| J_b | Bolt Mass Moment of Inertia |
| J_s | Solidworks Approximated Rotor Mass Moment of Inertia |
| J_m | Mathematical Approximated Rotor Mass Moment of Inertia |
| L | Motor Inductance |
| R | Motor Phase Resistance |
| R_{LL} | Motor Line-Line Resistance |

3.1 Motor Resistance

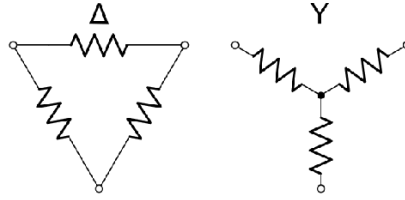


Figure 3.1: Motor Connection Configuration

BLDC motors are typically connected in two wiring configurations: WYE (Y) or delta (Δ), which can be seen in Figure 3.1. The RCTIMER GBM2804 utilizes the WYE (Y) configuration and will be analyzed as such. Due to the wiring of WYE systems, the neutral connection is typically unavailable for measurement on most motors. As a result, it is common to measure resistance by a line-line reading; however, in terms of motor control, it is the phase resistance and not the line-line resistance that is of importance. Converting between the phase and line-line resistance is quite simple and can be done by dividing the line-line resistance by two (Equation 3.1).

$$R = \frac{R_{LL}}{2} \quad (3.1)$$

3.1.1 Resistance Estimation

In order to gather a good estimate for the phase resistance of the RCTIMER GBM2804 motor, the resistance was measured line-line across all three phases. Each set of motor leads were hooked up to a digital multimeter, and the values were tabulated for each phase component in Table 3.2. An average was then calculated between the different line-line resistances to get the overall resistance of the motor.

Table 3.2: Measured motor resistance

| A-B | A-C | B-C |
|-----------------|--------------|--------------|
| 10.8 Ω | 6.2 Ω | 6.0 Ω |
| Average: | | 3.8 Ω |

Using Equation 3.1, it is then possible to find the overall phase resistance of the motor.

$$R = 3.8\Omega$$

3.2 Motor Torque Constant and Back EMF

The motor torque constant is a common parameter used in BLDC motors. It relates the armature current to the torque produced by a motor: $T = k_T i$. Many methods exist to determine the torque constant, including relating the motor velocity constant k_v , which is inversely related to the torque constant by $k_T = \frac{1}{k_v}$, or by measuring the line-line back-emf voltage per phase ($K_{e,p}$). $K_{e,p}$ is the peak value of the back-emf per angular velocity measured from line-neutral. However, since line-neutral is typically unavailable on most BLDC motors, the back-emf constant is often represented as a line measurement, K_e . The overall torque constant can then be related to the line measurement back-emf voltage for sinusoidal type outputs by Equation 3.2 or for trapezoidal outputs by Equation 3.3. [6].

$$k_T = \frac{\sqrt{3}}{2} K_e \quad (3.2)$$

$$k_T = K_e \quad (3.3)$$

However, with what will be shown in Chapter 4, in the mathematical model development for the NERMLAB, a dq rotor reference frame is used for the formulations. This results in

different motor torque and back emf constants, which both depend on the rotor flux linkage, ψ_R . Equations 3.4 and 3.5 can be used to calculate these new constants [6–8], and should be noted that an upper case E and lower case t are used, respectively, to denote the new dq reference frame. Likewise, in Equation 3.4, P denotes the number of stator poles that exist in the brushless motor, which is 12 in the case of the RCTIMER GBM2804. K_E also has units of *Webers · turns*, which is equivalent to the units of K_e , $\frac{V \cdot s}{rad}$, in a brushless motor [6].

$$k_E = \psi_R = \frac{2}{P} \sqrt{\frac{2}{3}} K_e \quad (3.4)$$

$$k_t = \frac{3}{2} \psi_R = \frac{3}{2} K_E \quad (3.5)$$

Because K_e can be experimentally determined, it is possible to find all of the motor constants for the RCTIMER GBM2804 using the above equations. One simply needs to measure the line-line sinusoidal or trapezoidal back-emf voltage at various speeds to get a good estimate of K_e . Care should be given when using the above approximations, 3.4 and 3.5, as these equations do not assume large current values [7].

3.2.1 Estimating the Back EMF Constant

In order to calculate the back-emf of the RCTIMER GBM2804 BLDC motor, an experiment had to be set up to measure the voltage generated by the motor. Three pieces of equipment were needed: an oscilloscope, the Motorlab, and a torque transmission shaft. The torque transmission shaft was a 3D printed part¹ that allowed the Motorlab to spin the RCTIMER GBM2804 at a constant speed to generate a back voltage. A line-line voltage (peak) was then read from the leads of the RCTIMER GBM2804 by an oscilloscope². This set up was run three times across the three motor phases to calculate an average back emf constant. The averaged data collected are tabulated in table 3.3.

¹Appendix B Figure B.3

²Appendix F Figure F.1

Table 3.3: Measured back voltage

| Average Speed ω_m (rad/s) | Average Peak Voltage (V) |
|----------------------------------|--------------------------|
| 36.47 | 1.87 |
| 43.56 | 2.24 |
| 50.84 | 2.54 |
| 65.66 | 3.36 |
| 72.93 | 3.68 |

There is a fairly linear relationship between the peak voltage and speed. Due to this fact, K_e can be approximated from the slope of $\frac{V}{\omega_m}$. The normal equation from the least-squares method was employed to find the best fit for the data in Table 3.3. Two matrices were constructed from the data, namely \mathbf{V} and $\boldsymbol{\omega}_m$.

$$K_e = (\boldsymbol{\omega}_m \boldsymbol{\omega}_m^T)^{-1} \boldsymbol{\omega}_m \mathbf{V}^T \quad (3.6)$$

From Equation 3.6, the back-emf constant was found to be:

$$K_e = 0.05 \frac{V \cdot s}{rad}$$

To verify that K_e was the best fit to that data, K_e was plotted against the collected data in Figure 3.2.

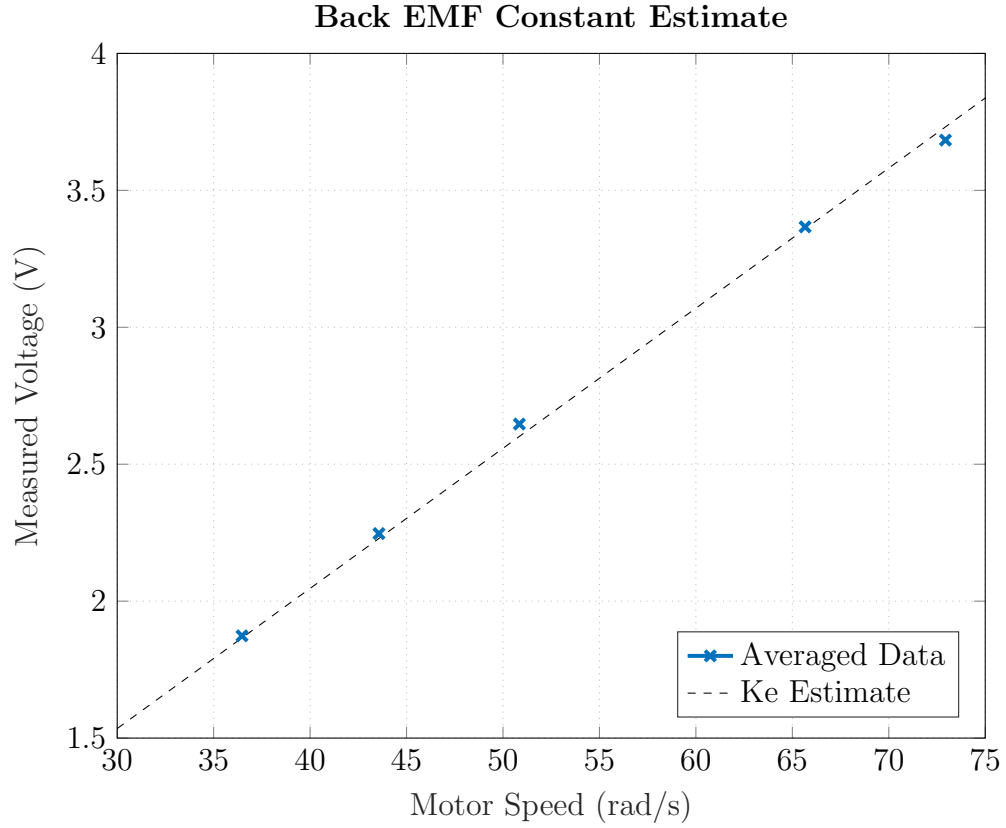


Figure 3.2: Measured Back EMF vs Speed

Using the equations developed in this section, the constants for the NERMLAB are tabulated below:

$$k_T = 0.04 \quad \frac{N \cdot m}{A} \quad [Sinusoidal]$$

$$k_T = 0.05 \quad \frac{N \cdot m}{A} \quad [Trapezoidal]$$

$$K_E = 0.007 \quad \text{Webers} \cdot \text{turns}$$

$$k_t = 0.01 \quad \frac{N \cdot m}{A}$$

3.3 Mass Moment of Inertia Estimation

Mass moment of inertia (J) is the equivalent to mass in a rotational system (commonly referred to as angular mass). More formally, it is defined as $J = \int r^2 dm$, where r is the distance to a mass from an axis of rotation.

The angular mass of the NERMLAB will be determined in two ways: approximating J through software modeling and approximating J through mathematical formulation.

Multiple inertias will be referenced throughout this thesis, one being the base load inertia of the rotor, which is comprised of individual neodymium magnets held by an outer aluminum casing, a second inertia being a simple steel washer, and finally, steel bolts that lock down the washers to the rotor. Table 3.4 tabulates the various parameters that are referenced in all mass moment of inertia calculations.

Table 3.4: Measured Inertia Parameters

| Parameter | Value |
|-----------------------|----------|
| Magnet Thickness | 0.002 m |
| Rotor Outer Radius | 0.0164 m |
| Rotor Mass | 0.018 kg |
| Washer Mass | 0.045 kg |
| Washer Inner Radius | 0.0105 m |
| Washer Outer Radius | 0.0253 m |
| Bolt Mass | 0.005 kg |
| Bolt Radius | 0.0082 m |
| Bolt Off Center Dist. | 0.0085 m |

3.3.1 Software Modeling of Mass Moment of Inertia

Computer Aided Design (CAD) software was utilized to construct a 3-D model of the neodymium magnets and aluminum casing. CAD software like SolidWorks has the ability to determine complex mass moment of inertias via numerical methods. Knowing the average density of neodymium ($7.3 - 7.5 \frac{g}{cm^3}$) and aluminum ($2.7 \frac{g}{cm^3}$), it is possible to numerically find an inertia (J_s) of the rotor.

$$J_s = 4.0842 \times 10^{-6} \text{ kg} \cdot \text{m}^2$$

3.3.2 Mathematical Approximation of Mass Moment of Inertia

To simplify the mathematical analysis of the mass moment of inertia calculation of the angular mass of the NERMLAB, an engineering assumption will be made that the angular mass is a rotating ring mass. This assumption is valid for the particular motor used in this thesis, due to the fact that most of the mass is concentrated around the outside perimeter of the motor. The outside ring mass of the motor contributes the most to the inertial load, so the mathematical formulation would result in the following equation:

$$J_z = mr^2 \tag{3.7}$$

Knowing the outer radius of the rotor, it is possible to find the mathematical approximation of the rotor's inertia using equation 3.7.

$$J_m = 4.8413 \times 10^{-6} \text{ kg} \cdot \text{m}^2$$

Washer Inertia

Approximating the washer inertia is relatively straight forward as the geometry is simple to measure. As stated in Section 3.3.2, the only dimensions that need measuring are the inner

and outer radii, which are provided in table 3.4.

$$J_w = \frac{m}{2}(r_i^2 + r_o^2) \quad (3.8)$$

With the measured radii results and mass properties from Table 3.4, the washer inertia can be easily approximated using Equation 3.8, which is the formula for a thick-walled cylinder.

$$J_w = 1.7 \times 10^{-5} \text{ kg} \cdot \text{m}^2$$

Bolt Inertia

The bolt inertia is slightly more complex to calculate than the washer inertia since the bolts lay off axis from the center of rotation. However this can be easily approximated using the parallel axis theorem, Equation 3.9, where d is the distance to the center of rotation. Likewise, the bolt will be approximated as a solid cylinder, as stated by Equation 3.10, where $J_{b,o}$ is the inertia at the origin.

$$J = J + md^2 \quad (3.9)$$

$$J_{b,o} = \frac{1}{2}mr^2 \quad (3.10)$$

Combining Equations 3.9 and 3.10 results in Equation 3.11, which can be used to calculate the total inertia of the bolts.

$$J_b = J_{b,o} + md^2 \quad (3.11)$$

$$J_b = 5.3 \times 10^{-7}$$

3.3.3 Lumped Mass Moment of Inertia

$$J_r = \frac{J_m + J_s}{2} \quad (3.12)$$

Having two approximations for the inertia of the rotor, it is now possible to calculate an average between the two. Using the results from Sections 3.3.2 and 3.3.1 and Equation 3.12, the overall rotor inertia is found to be:

$$J_r = 4.5 \times 10^{-6} \text{ kg} \cdot \text{m}^2$$

For most of the experiments ran in this thesis, additional inertia is used to slow down the response of the system to allow a better visualization of what is happening when looking directly at the motor as it runs. For the total lumped inertia of the system, three steel washers and four loading locking bolts are being used. Since all inertias in the system are rotating about the same axis, the total lumped inertia of the system is described by Equation 3.13.

$$J = 3J_w + J_r + 4J_b \quad (3.13)$$

$$J = 5.762 \times 10^{-5} \approx 5.8 \times 10^{-5} \text{ kg} \cdot \text{m}^2$$

3.4 Motor Inductance

The motor inductance is measured by building a simple circuit, which consists of a resistor in series with one of the motor's phase circuitry, as depicted in Figure 3.3.

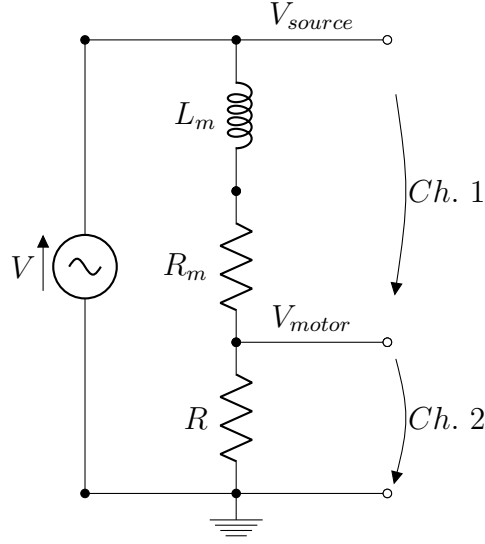


Figure 3.3: Inductance Measurement Circuit

A function generator is required for this experiment to generate a variable frequency sinusoid across the external resistor and motor. Knowing that Figure 3.3 is a simple voltage divider, it is possible to arrive at Equation 3.14 using the relationship $|V_{motor}| = \frac{1}{2}|V_{source}|$, where f is the driving frequency of the sinusoid. Then, the voltages of two outputs are monitored on separate channels on an oscilloscope, and the sinusoidal frequency is increased until the motor output's voltage reaches one half of that of the source voltage.

$$L = \frac{R\sqrt{3}}{2\pi f} \quad (3.14)$$

3.4.1 Inductance Estimate

With the outlined procedure developed in the previous Section (3.4), three resistors were measured, and the voltages across each junction were tabulated when an appropriate frequency was reached. Table 3.5 hosts the experimental results, along with the calculated average inductance that was found from Equation 3.14.

Table 3.5: Inductance Experiment

| Resistance (Ω) | V_{source} (V) | V_{motor} (V) | Frequency (kHz) | Inductance (H) |
|-------------------------|------------------|-----------------|-----------------|-----------------------|
| 22 | 0.392 | 0.176 | 8.34 | 7.27×10^{-4} |
| 98.4 | 1.04 | 0.507 | 34.4 | 7.88×10^{-4} |
| 461 | 1.88 | 0.920 | 167.5 | 7.58×10^{-4} |
| Average: | | | | 7.58×10^{-4} |

$$L = 7.58 \times 10^{-4} \text{ H}$$

3.5 Viscous Friction

Section 3.5 will not detail the process conducted to determine the viscous friction of the NERMLAB, as this will be discussed in Chapter 4, where a laboratory has already been developed for students to estimate this coefficient. For the sake of completeness the result will be provided below.

$$b = 3.0 \times 10^{-4} \text{ N} \cdot \text{m} \cdot \text{s}$$

Chapter 4

Development of Mathematical Models for NERMLAB

Chapter 4 will develop various mathematical models for the NERMLAB and Motorlab. These models are used throughout this thesis to help compare, analyze, and develop effective control solutions for motor controllers. Rather than have each experiment develop its own model in the corresponding chapter, it will be done here to help simplify the content of each experiment. Since the NERMLAB and Motorlab use different input sources ¹, different mathematical models will be developed for each system separately.

4.1 Brushless Motor Theory

Section 4.1 briefly covers brushless motor theory in an aim to give a background to the reader for the proceeding sections that use it in model development. Since this is a short introduction to the topic, many aspects of the theory are not mentioned, as it is outside the scope of this thesis, but the reader can reference an excellent masters report by James Mevey at Kansas State University, Sensorless Field Oriented Control of Brushless Permanent Magnet Synchronous Motors [6].

¹Nermlab uses voltage control. Motorlab uses current control.

Brushless motors differ from their counter parts, brushed DC motors, in the fact that they require digital or electronic logic to control the motor current and voltage to produce motor commutation. It is the commutation process that is complicated, as the models and mathematical formulations tend to be fairly involved, and as a result, require special transforms. The space vector is one such transform and is a linear transformation of three phase variables [6]. It's purpose is to analyze three-phase circuits, which gives it great use in brushless motor theory. There are two ways at looking at the space vector, through a stationary $\alpha\beta$ reference frame, or a rotating synchrononous dq frame. On the NERMLAB, the rotating synchrononous frame is used first as a voltage command from the controller. However, in order to control the physical motor, it is necessary to transform from the rotating frame back to the original three phase system. One way of carrying out this process is through the use of the inverse Clarke and Park transforms, Equations 4.1 and 4.2 respectively. In Equation 4.1, k is a scaling factor, which in the case of the NERMLAB is $\sqrt{\frac{2}{3}}$, which yields a power-invariant form, which is more formally known as the Concordia transform.

$$\mathbf{x}_{\alpha\beta} = \mathbf{P}^{-1}\mathbf{x}_{dq} \leftrightarrow \begin{bmatrix} x_{\alpha} \\ x_{\beta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x_d \\ x_q \end{bmatrix} \quad (4.1)$$

$$\mathbf{x}_{abc} = \frac{1}{k}\mathbf{C}^{-1}\mathbf{x}_{\alpha\beta} \leftrightarrow \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} = \frac{1}{k} \begin{bmatrix} \frac{2}{3} & 0 \\ -\frac{1}{3} & \frac{1}{\sqrt{3}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_{\alpha} \\ x_{\beta} \end{bmatrix} \quad (4.2)$$

4.2 Electrical Dynamics

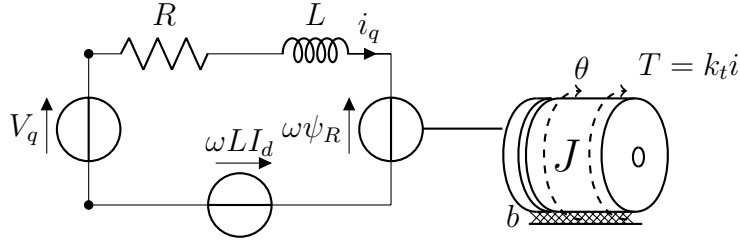


Figure 4.1: Electrical and Mechanical Diagram of NERMLAB

Figure 4.1 hosts an electro-mechanical model for the NERMLAB. Here, it is important to note that this diagram only consists of the quadrature axis, q , of the dq model. This is a result of the fact that the NERMLAB only uses the V_q voltage as its input to the motor (e.g. $V_d = 0$), allowing for the model to be simplified [7].

Using Kirchoff's voltage law, it is possible to find the dynamics of figure 4.1.

$$\sum V = 0$$

$$V_q(t) = Ri_q(t) + \omega LI_d + L \frac{di_q}{dt} + \omega \psi_R \quad (4.3)$$

Equation 4.3 can be simplified to produce a dynamic system that is easier transform into the models that will be used in this thesis. Note that ω is the angular velocity of rotation, and not the electrical angle rate of the motor. The following assumptions will be made:

1. Because the pole at $-\frac{R}{L}$ ($6596.3 \frac{rad}{s}$) is ten times as large as the other dynamics of the system, $L \frac{di_q}{dt}$ can be ignored in the analysis process.
2. The term ωLI_d can be neglected as $V_d = 0$.
3. $\psi_R = K_E$ in equation 4.4 since ψ_R was calculated in Chapter 2 using K_e . Note that ψ_R is the rotor flux linkage.

With these simplifications, Equation 4.3 can then be reduced to Equation 4.4. Since the derivations that take place in this thesis consist only of the quadrature axis, the subscripts q will be dropped from the variables throughout the remainder of the text.

$$V(t) = Ri(t) + \omega K_E \quad (4.4)$$

The torque the motor produces can be related to the supplied current from the motor driver, i_q , as in equation 4.5.

$$T = k_t i(t) \Leftrightarrow i(t) = \frac{T(t)}{k_t} \quad (4.5)$$

Using equations 4.5 and 4.4, the motor torque and current Equations, 4.6 and 4.7, can be derived respectively.

$$T(t) = \frac{k_t}{R}(V(t) - K_E \omega) \quad (4.6)$$

$$i(t) = \frac{V(t) - K_E \omega}{R} \quad (4.7)$$

4.3 Combined Dynamics - Electrical and Mechanical

Section 4.3 will detail the model development for the various electro-mechanical models that are used throughout this thesis. Sections 4.3.1 and 4.3.2 will derive mathematical models for position and speed systems, respectively, with different input sources. For the model derivations using current as an input source, the closed-loop current control system is assumed to be much faster than the mechanical dynamics. As a result of this assumption, only the mechanical dynamics and controller will be in the model development.

4.3.1 Position Models

Current as Input Source

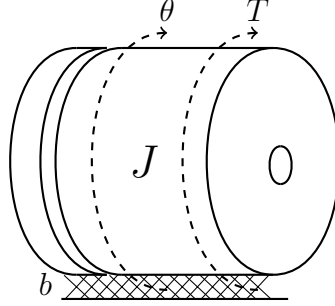


Figure 4.2: Position Model

The best way to start the formulation is to begin with a time domain differential equation of the mechanical system. Because the system is composed of only an angular mass and viscous friction (figure 4.2), a describing differential equation can be written as such.

$$T = k_T i(t) = b\dot{\theta}(t) + J\ddot{\theta}(t) \quad (4.8)$$

Taking the Laplace transform of equation 4.8:

$$k_T I(s) = (bs + Js^2)\theta(s) \quad (4.9)$$

the transfer function can then be developed for G_m from equation 4.9.

$$G_m(s) = \frac{\theta(s)}{I(s)} = \frac{k_T}{Js^2 + bs} \quad (4.10)$$

Voltage as Input Source

Equation 4.10 adequately describes the system for the Motorlab because the electrical dynamics are much faster than the mechanical. However, in the case of the NERMLAB, voltage control is used, and as a result, a different model must be developed.

Starting with the differential equations of the electrical dynamics and mechanical dynamics, the following equations are found.

$$V(t) = Ri(t) + L\frac{di}{dt} + K_E\dot{\theta}(t) \quad (4.11)$$

$$T(t) = J\ddot{\theta}(t) + b\dot{\theta}(t) \quad (4.12)$$

Taking the Laplace transform of equations 4.11 and 4.12.

$$V(s) = RI(s) + LsI(s) + K_Es\theta(s) \quad (4.13)$$

$$T(s) = (Js^2 + bs)\theta(s) \quad (4.14)$$

From section 4.2, the relationship between torque and current is known. Substituting equation 4.5 into equation 4.14, an equation for current is found.

$$I(s) = \frac{(Js^2 + bs)}{k_t}\theta(s) \quad (4.15)$$

Substituting equation 4.7 into equation 4.13, grouping and finding a common denominator in the process, yields equation 4.16.

$$\frac{\theta(s)}{V(s)} = \frac{k_t}{(Ls + R)(Js^2 + bs) + K_Ek_ts} \quad (4.16)$$

As stated in section 4.2, the electrical dynamics involving L are much larger than any other dynamics in the system; therefore, L can be ignored, resulting in the final position equation in terms of voltage as an input source.

$$\frac{\theta(s)}{V(s)} = \frac{k_t}{RJ s^2 + (Rb + K_Ek_t)s} \quad (4.17)$$

4.3.2 Speed Models

Current as Input Source

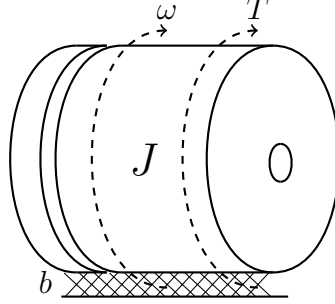


Figure 4.3: NERMLAB Speed Model

The position models just developed in section 4.3.1 can be used to help derive models for speed. The process is a relatively straight forward one, with one simple substitution needed. Knowing the relationship between speed and position, it is possible to write the following equation.

$$\theta(s) = \frac{\omega(s)}{s} \quad (4.18)$$

Substituting equation 4.18 into 4.10 simply cancels the free integrator, leaving the final equation in terms of speed.

$$\frac{\omega(s)}{I(s)} = \frac{k_T}{Js + b} \quad (4.19)$$

Voltage as Input Source

Just as in section 4.3.2, a simple substitution of equation 4.18 into 4.17 yields the final speed model in terms of voltage.

$$\frac{\omega(s)}{V(s)} = \frac{k_t}{RJ s + (Rb + K_E k_t)} \quad (4.20)$$

4.3.3 Low-Pass Filter

A low-pass filter is utilized on the NERMLAB when a speed control system is used. This low-pass filter is simply a second order system with a cut off frequency at $300 \frac{rad}{s}$, and damping ratio of $\frac{1}{3}$, as seen in equation 4.21 and figure 4.4.

$$G_{hf} = \frac{300^2}{s^2 + 212s + 300^2} \quad (4.21)$$

Bode Diagram

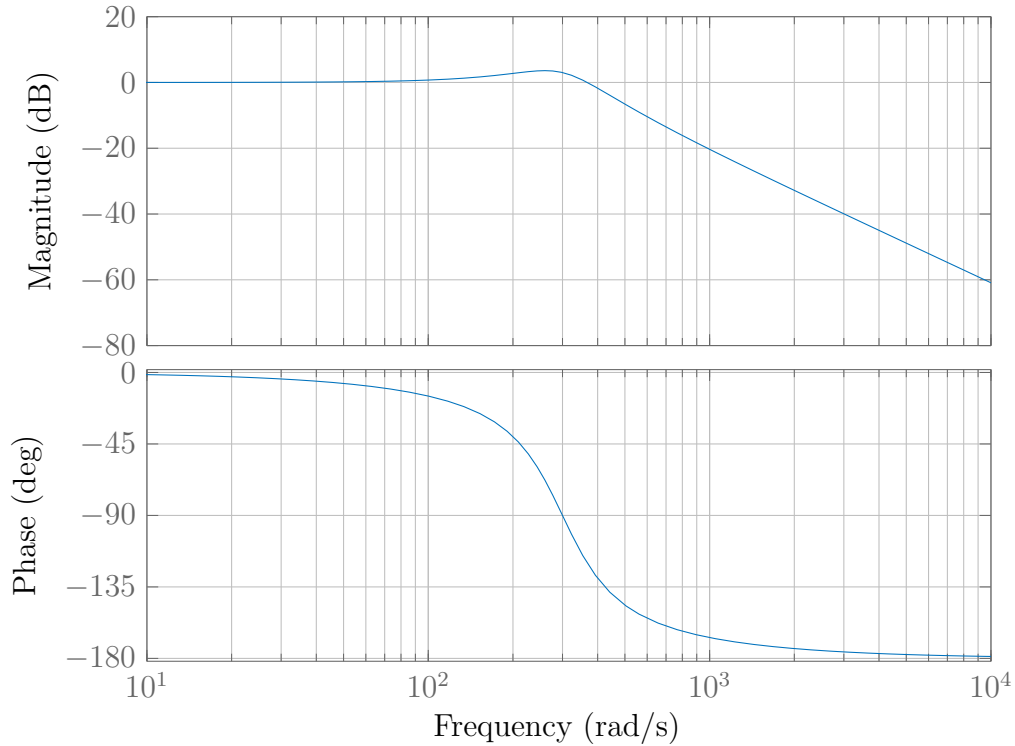


Figure 4.4: 2nd order low-pass filter with a cutoff frequency of $300 \frac{rad}{s}$

The low-pass filter on the NERMLAB is used to attenuate high frequency noise² that occurs due to the quantization of the encoder readings when converting angular position to speed, which is accomplished through taking a derivative. The low-pass filter assists in smoothing the signal of the output (speed), to minimize this high frequency noise.

²Higher frequency noise than that of the cutoff frequency

Chapter 5

Approximating Friction of a BLDC Motor

One of the important aspects of control theory is predicting a systems behavior with mathematical models. Because of this fact, friction, in most circumstances, needs to be included as a part of the developed model to accurately predict a systems response. In motor applications, there are various types of friction that come into play, such as: static, viscous, Stribeck, and Coulombic friction. However, in the case of this experiment, and for the sake of model simplicity, Stribeck and Coulombic friction are ignored, with focus geared towards static and viscous friction. A fit will be made between these two types of friction forces, which will help develop an overall friction coefficient (b) to be used in mathematical formulations and control analysis. Results from this chapter will then be compared against the Motorlab. The laboratory that students conduct in Control of Mechanical Systems I at Kansas State University is outlined in Appendix C and will be the guideline for the experiment in Chapter 5.

5.1 Friction Estimate

The NERMLAB makes use of a voltage controller for its input. Therefore, effects like back-emf and motor resistance must be included in the model. Chapter 4 developed the open-loop mathematical speed model for the NERMLAB and will be used in this section. Equation 4.20, as well as, the describing differential equation of the mechanical model of the NERMLAB (Equation 4.8) are restated below.

$$\frac{\omega(s)}{V(s)} = \frac{k_t}{RJ s + (Rb + K_E k_t)}$$

$$T = b\dot{\theta}(t) + J\ddot{\theta}(t)$$

What can be seen from equation 4.8 is that if a constant torque (voltage) is applied to the NERMLAB and is allowed to achieve steady state ($\ddot{\omega}(t) = 0$), then equation 4.8 reduces to:

$$T(t) = b\dot{\omega}(t) \tag{5.1}$$

What this means is that in order to predict a friction coefficient for the NERMLAB, all that needs to be done is to supply a voltage to the motor until steady state is achieved. Then an average value for the recorded speed can be tabulated. There are two ways that friction can be estimated with this set-up:

1. Plot Torque vs Speed and find a suitable best fit for the slope. This slope will accurately estimate the friction coefficient. However, because voltage control is being used on NERMLAB, equation 4.6 must be used to convert the voltage input and speed value to a torque.
2. Plot Voltage vs Speed and find a suitable best fit for the slope. Unlike the first item, the slope does not directly give the friction coefficient. Rather, the DC gain of equation

4.20 must be found. From there, knowing that the DC gain is the best fit slope value, solving equation 5.2 for b (eq. 5.3) can accurately estimate a friction coefficient. Here K_{DC} was found to be 12.2×10^{-2} , which results in a friction value of $3.0 \times 10^{-4} \frac{N \cdot m \cdot s}{rad}$, predicted by equation 5.3.

$$K_{DC} = \frac{k_t}{K_E k_t + Rb} \quad (5.2)$$

$$b = \frac{k_t - (1/K_{DC})K_E k_t}{(1/K_{DC})R} \quad (5.3)$$

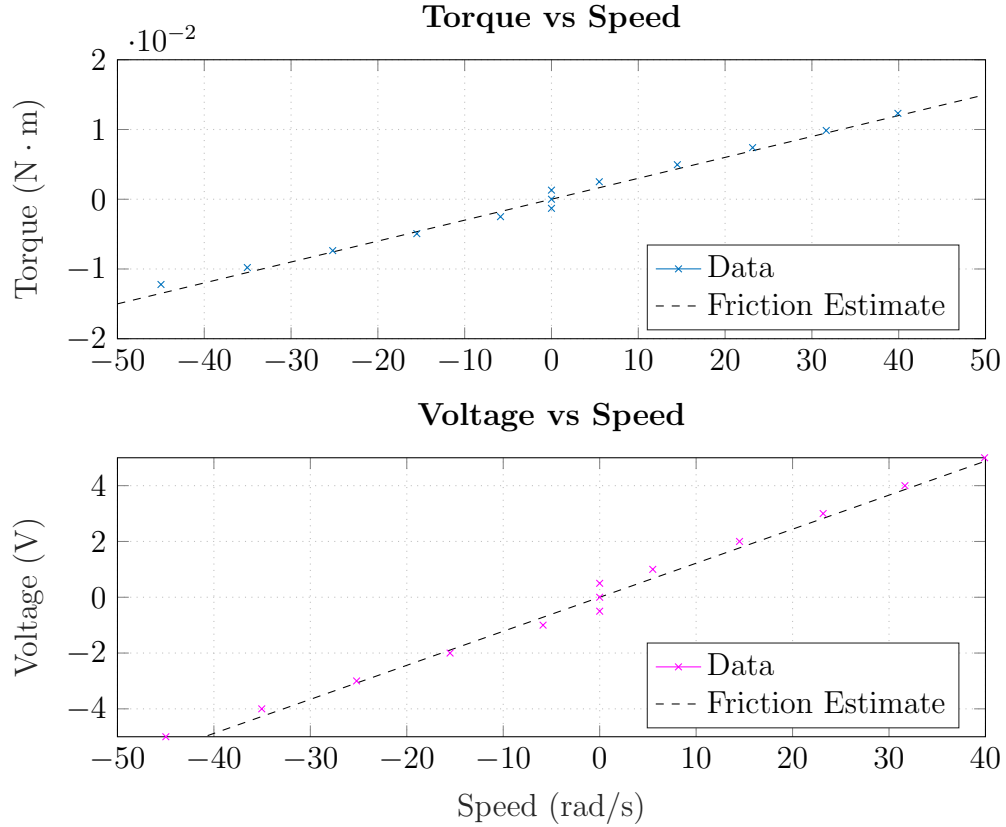


Figure 5.1: Viscous friction estimate with voltage vs speed and torque vs speed

A condensed list of the results from the experiment are tabulated in Table 5.1, while the full data set can be found in Appendix F, Table F.1. Both items listed above were carried out,

and a friction coefficient for both experiments returned the same value of $3.0 \times 10^{-4} \frac{N \cdot m \cdot s}{rad}$.

Table 5.1: Friction Experiment

| Parameter | | | | | | | |
|----------------|---------|---------|---------|---|--------|--------|--------|
| Voltage (V) | -5 | -3 | -1 | 0 | 1 | 3 | 5 |
| Speed (rad/s) | -45.0 | -25.2 | -5.8 | 0 | 5.5 | 23.2 | 40.0 |
| Torque (N·m/s) | -0.0122 | -0.0074 | -0.0025 | 0 | 0.0025 | 0.0074 | 0.0099 |

5.2 Speed Decay

Now that a value for friction is found, it is possible to conduct an additional experiment doing an initial condition response. The initial condition response is a good way of showing how experimental data can follow simple theoretical models. A way of doing this on the NERMLAB is to command a constant voltage so as to get a steady state speed on the rotor. Then, the voltage on the motor is turned off, letting the motor's speed decay to zero.

To find a describing differential equation of the speed decay, it is necessary to start with equation 4.8. From equation 4.8, taking the Laplace transform with an initial speed condition results in the following equation:

$$T(s) = 0 = Js\omega(s) + b\omega(s) - J\omega(0) \quad (5.4)$$

Here, torque is set to zero since the motor was given an initial velocity, and the voltage was set to zero at time zero. From equation 5.4, it is possible to simplify further, arriving at:

$$\omega(s) = \omega_0 \frac{J}{Js + b} \quad (5.5)$$

From equation 5.5, performing the inverse Laplace will give a time domain solution that can be used to compare the experimental results too.

$$\omega(t) = \omega_0 e^{-\frac{b}{J}t} \quad (5.6)$$

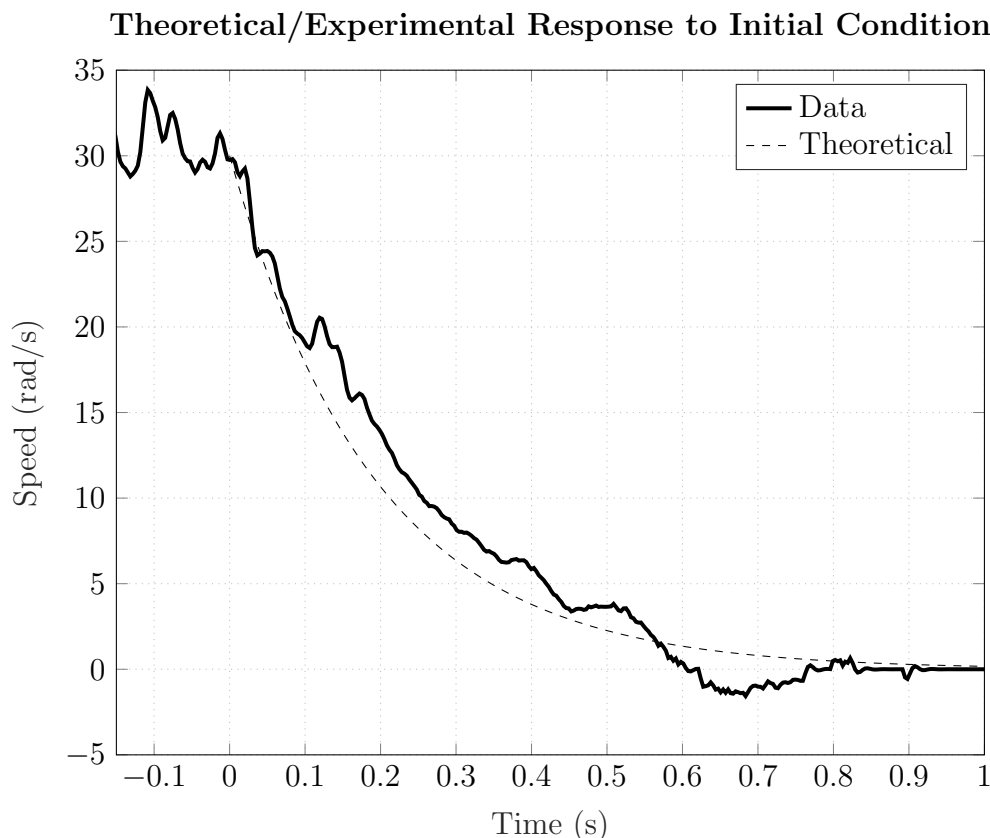


Figure 5.2: Speed decay of an initial condition (NERMLAB)

Here, Figure 5.2 shows a good fit to the theoretical model developed by equation 5.6. At lower speeds, Figure 5.2 indicates that the higher coefficient of static friction starts to dominate the dynamics of the system, resulting in quicker decay than predicted by the theoretical model.

5.3 Motorlab Results

The friction experiment for the Motorlab is exactly the same as the one conducted for the NERMLAB, with the only difference being that instead of commanding a voltage, the Mo-

torlab uses current. A result of using current over voltage is that it simplifies the laboratory a bit more. For the NERMLAB, the voltage needed to be converted to a torque using equation 4.6, whereas the Motorlab simply needs the commanded current to be multiplied by the torque constant, k_T , to obtain the corresponding torque. Referencing figure 5.3, it can be seen that the Motorlab does a good job at demonstrating how a simple model can adequately describe a resulting response, in this case an exponential decay. The Motorlab does a better job at showing how static friction dominates at lower speeds, but both systems show a good relationship between the theoretical models and the actual data.

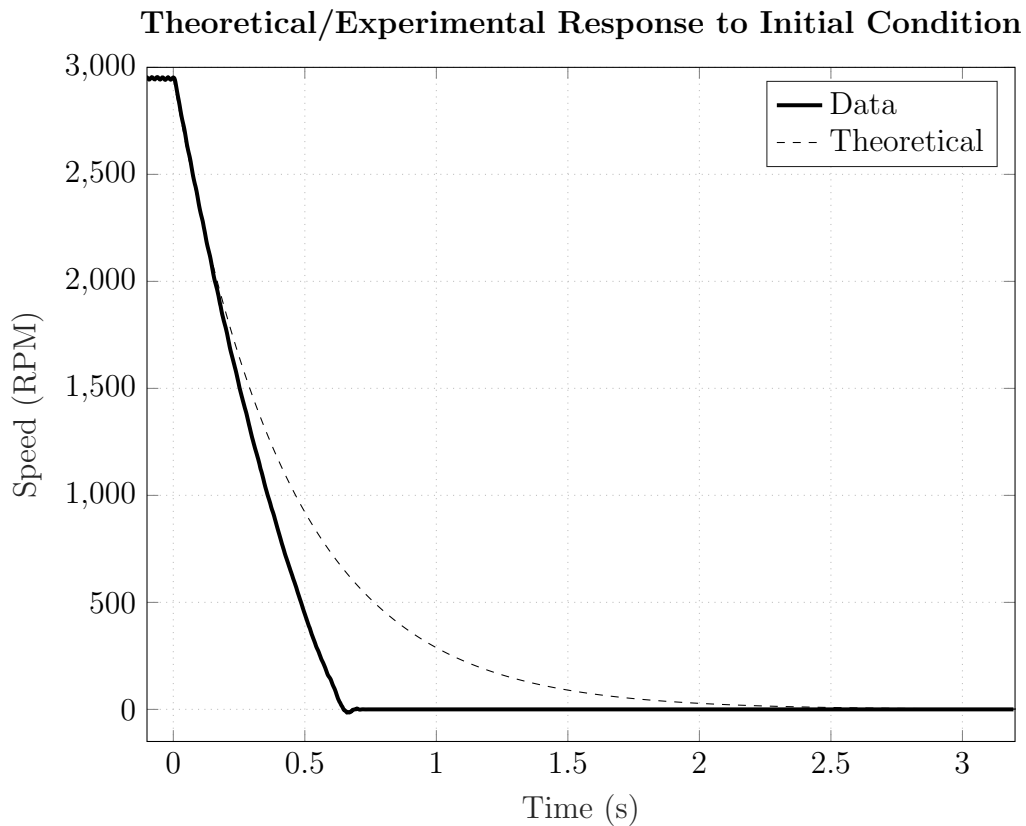


Figure 5.3: Speed decay of an initial condition (Motorlab)

Chapter 6

Frequency of Oscillation of a Position Control System

Chapter 6 will discuss an experiment with a position control system on the NERMLAB. The main purpose of this experiment is not to develop a better position control system, but rather, demonstrate the concept of changing pole locations and a characterization of the responses to the changing gains and poles. In the case of both the NERMLAB and Motorlab system, the closed-loop solution of equation 4.17 and 4.10, respectively, results in two real poles in the s-plane, causing only increasing oscillation of the system with an increasing gain. This idea will be reinforced by looking at the frequency and decay rate of the oscillations in the various responses generated. Results produced by the NERMLAB will then be compared against the Motorlab. The mathematical open-loop position system was developed in chapter 4, equation 4.17. Since this experiment involves using a position control system, it is necessary to derive a closed-loop solution from the open-loop equation (eq. 4.17), which will be covered in the proceeding section 6.1. This experiment follows the procedure outlined in Laboratory 5 in Appendix C.

6.1 Mathematical Model of a Closed-Loop Position Control System

Figure 6.1 establishes a block diagram of the position control system used in this experiment. Here, G_c refers to the controller, which in the case of this chapter, is a simple proportional gain, K_p . G_m represents the plant model, which is comprised of the electromechanical dynamics (equation 4.17 restated as equation 6.1, below).

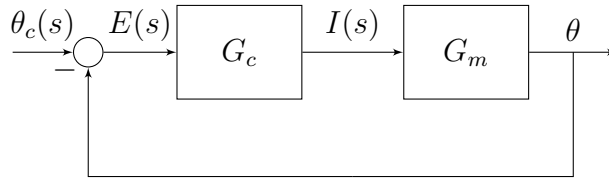


Figure 6.1: Closed Loop Control System

$$G_m = \frac{\theta(s)}{V(s)} = \frac{k_t}{RJ s^2 + (Rb + K_E k_t)s} \quad (6.1)$$

To find the closed-loop system, the blocks in series (G_c and G_m) must be reduced to a single block through simple multiplication of $G_c \cdot G_m$, which results in the following equation:

$$G = G_c \cdot G_m = \frac{K_p k_t}{RJ s^2 + (Rb + K_E k_t)s} \quad (6.2)$$

Assuming unity feedback, the final closed-loop solution can be developed through a simple feedback calculation, $\frac{G}{1+G}$:

$$T = \frac{\theta(s)}{\theta_c(s)} = \frac{G}{1+G} = \frac{K_p k_t}{RJ s^2 + (Rb + K_E k_t)s + K_p k_t} \quad (6.3)$$

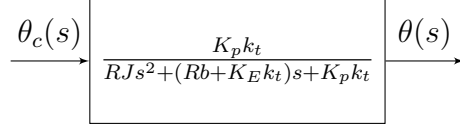


Figure 6.2: Block Diagram Reduction of Figure 6.1

In the case of the Motorlab system, the process is similar to the one carried out above. Using equation 4.10, the closed-loop position control system of the Motorlab is:

$$\frac{\theta(s)}{\theta_c(s)} = \frac{K_p k_T}{Js^2 + bs + K_p k_T} \quad (6.4)$$

6.2 Experiment

Three different proportional gains will be used in this experiment, listed in table 6.1. A commanded square wave will be the input voltage to the NERMLAB to simulate a step input to the system. Note that the magnitude of the square wave values in table 6.1 are different for each gain. This is a result of conducting experiments on real systems, as the NERMLAB has a saturation voltage of 8V. Since the units of K_p are $\frac{V}{rad}$, commanding a larger gain while statically defining an input position will result in saturation of the motor driver, resulting in inconsistent data that does not follow the developed model.

Table 6.1: Table of experimental gains to be used on the NERMLAB system

| Gain ($\frac{V}{rad}$) | Magnitude of Square Wave (rad) |
|--------------------------|-----------------------------------|
| 0.9 | 8.8 |
| 2.5 | 3.2 |
| 25 | 0.32 |

As stated in the beginning of this chapter, increasing the gain of the NERMLAB in a

position control mode should result only in increasing oscillations of the system, without improving the system's settling time. This phenomena occurs because the system poles only travel up and down, parallel to the $j\omega$ axis (depicted in figure 6.3), which in the s-plane causes a system's response to become more oscillatory with an increasing frequency of the imaginary part of the pole. This can be predicted by looking at the root locus of the open-loop system (eq. 4.10).

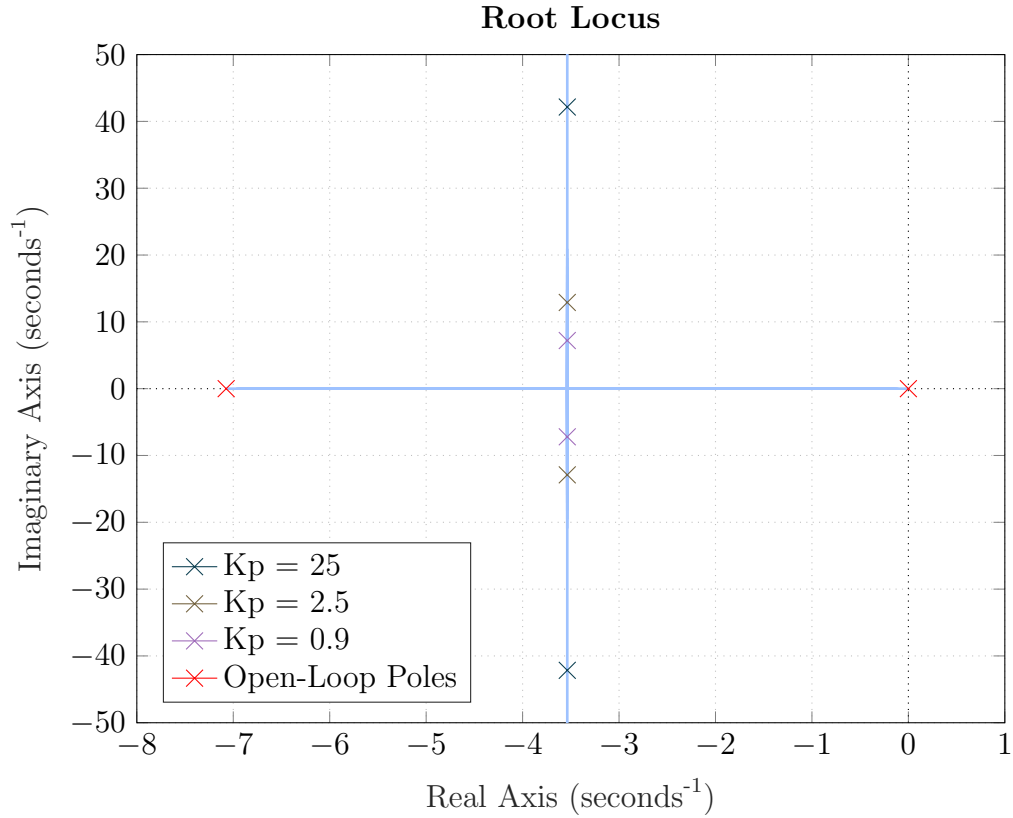


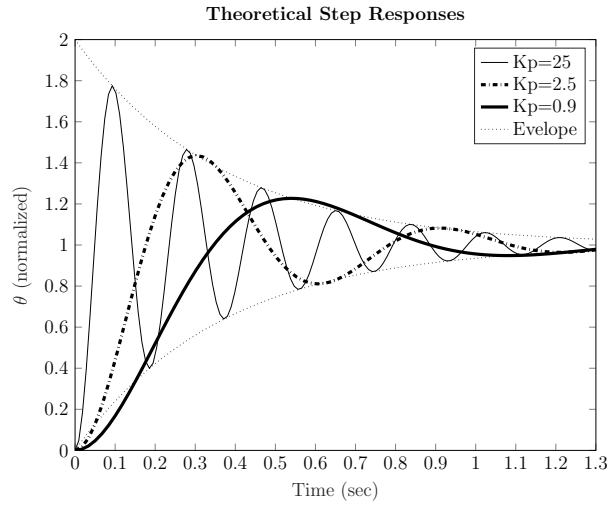
Figure 6.3: Root Locus of NERMLAB's position control system

Experimental data was collected for a set of three gains listed in table 6.1. The period of oscillations and the time constant were found graphically from the plotted data. In order to caculate the period of oscillations for the model, it is simply $\frac{2\pi}{\omega_d}$, where ω_d is the imaginary portion of the pole, $-\zeta\omega_n \pm \omega_d j$. Likewise, the theoretical time constant can be directly calculated from $\frac{1}{\zeta\omega_n}$.

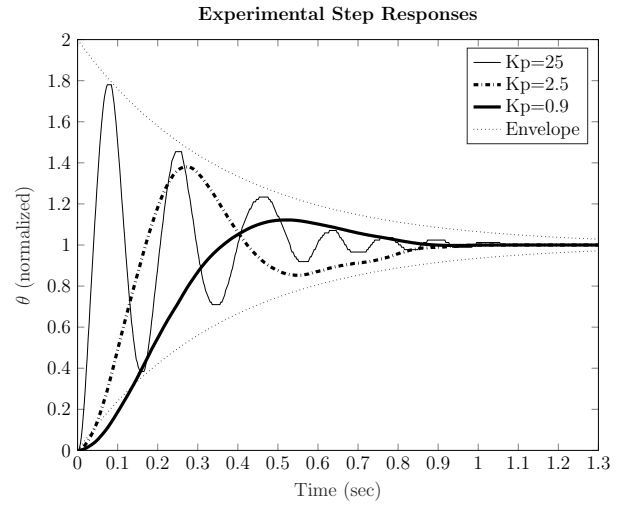
Table 6.2: NERMLAB results from the conducted experiment compared with a theoretical model

| Gain ($\frac{V}{rad}$) | Theoretical CL Poles ($\frac{rad}{s}$) | Theoretical Period of Oscillations (<i>seconds</i>) | Measured Period of Oscillations (<i>seconds</i>) | Theoretical Time Constant (<i>seconds</i>) | Measured Time Constant (<i>seconds</i>) |
|------------------------------------------|------------------------------------------------------------------|------------------------------------------------------------------------|---------------------------------------------------------------------|---------------------------------------------------------------|------------------------------------------------------------|
| 0.9 | $-2.74 \pm 5.8j$ | 1.1 | 0.95 | 0.365 | 0.365 |
| 2.5 | $-2.74 \pm 10.4j$ | 0.60 | 0.55 | 0.365 | 0.365 |
| 25 | $-2.74 \pm 33.8j$ | 0.18 | 0.17 | 0.365 | 0.365 |

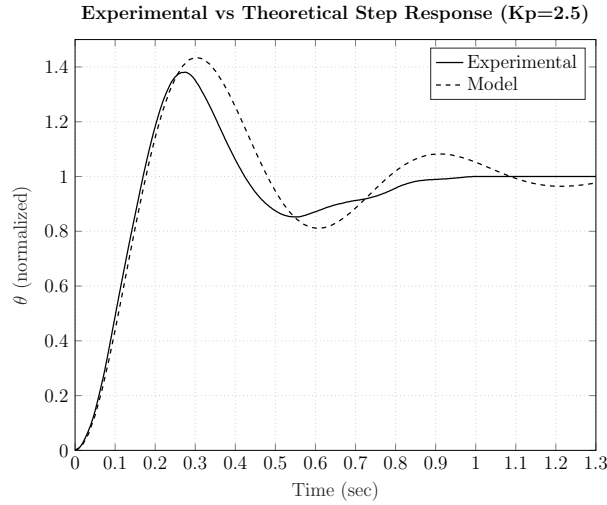
From table 6.2 and figure 6.4 it can be seen that there is a good match between what is observed in the real system with that of the model. Figures 6.4a and 6.4b show the step responses for the model and experiment, respectively. The effects of non-linear friction can be seen in figures 6.4c and 6.4d, as the oscillations damp out quicker than that predicted by the model.



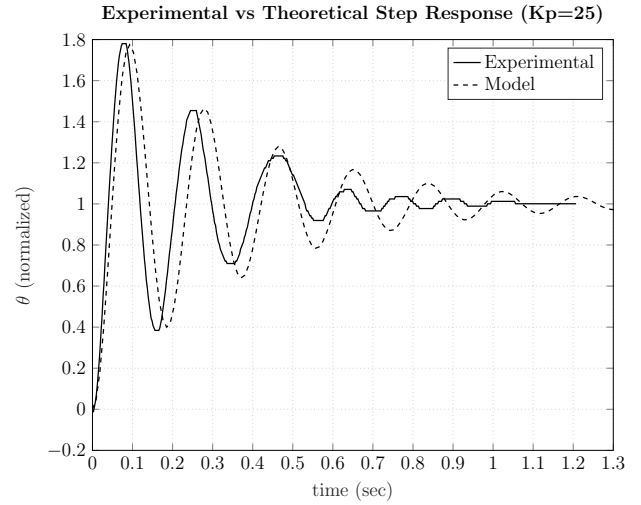
(a) Theoretical step responses



(b) Experimental step responses



(c) Comparison plot for $K_p = 2.5$



(d) Comparison plot for $K_p = 25$

Figure 6.4: Step responses that result from increasing proportional gain in a position control system for NERMLAB

It is also useful to demonstrate the model's validity by showing a larger range of gains. This helps prove to students, as long as saturation effects are taken into account, that developed models follow the same trends that the experimental results provide. In Figure 6.5, a lower gain of 0.2 is used show an overdamped response and how actual data compares to it. Likewise, a much larger gain of 40 is used to show how the data has far more damping

of the oscillations, due in part to friction, than the model predicts.

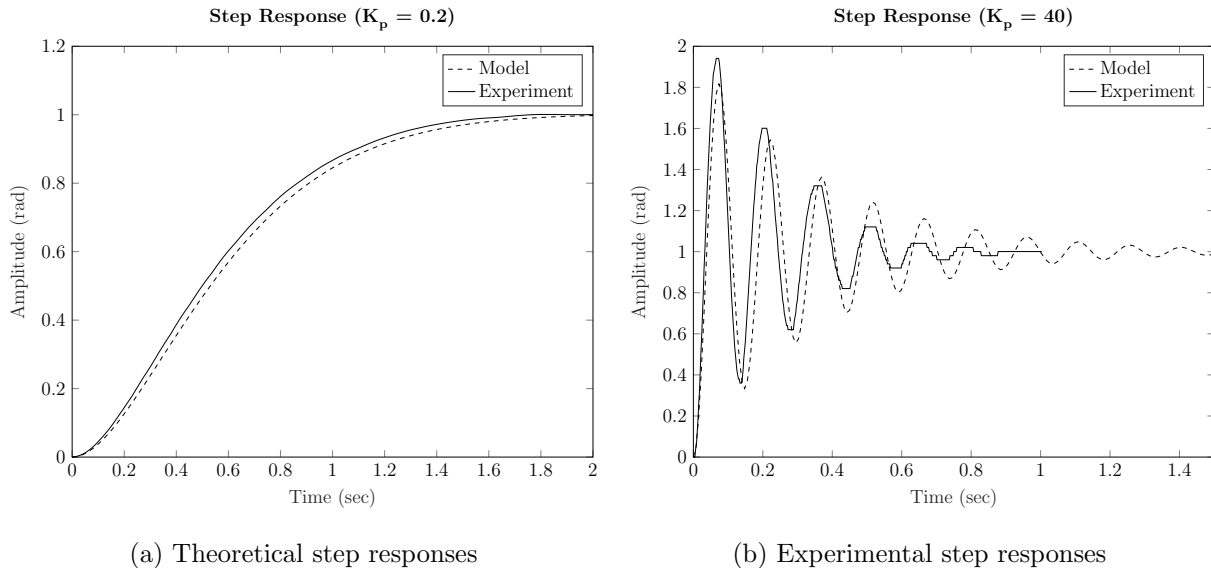
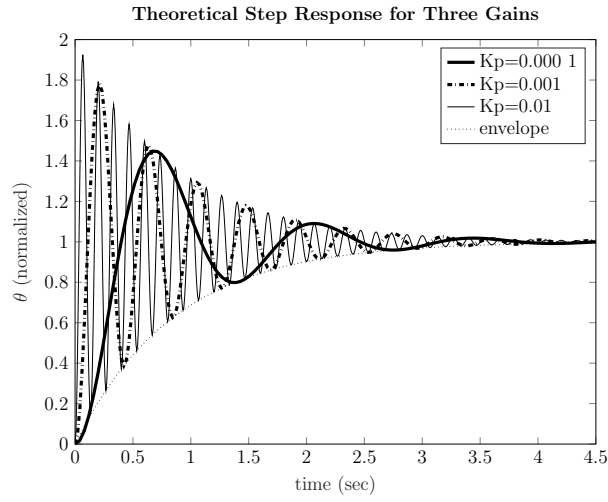


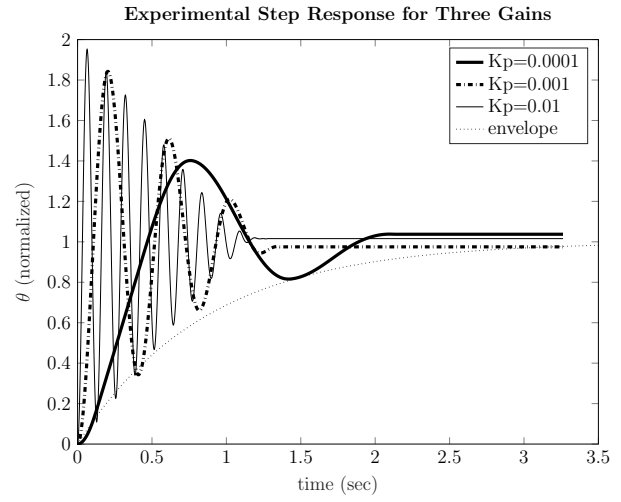
Figure 6.5: Step responses that result from gains outside the range conducted in the experiment, and how they compare to the developed models.

6.3 Motorlab Results

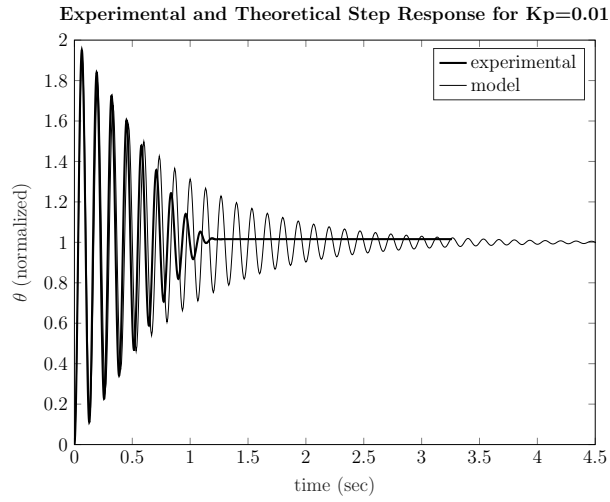
The Motorlab system performs slightly different than NERMLAB. From figure 6.6, the effects of nonlinear friction and saturation show up more prominently than they do for the NERMLAB. This causes the Motorlab to have more damped oscillations in its response, as can be seen in figure 6.6c. However, when a gain outside that of the experiment ($K_p = 40$) was used, these damped oscillations became more apparent on the NERMLAB. While it is important for students to be aware of other unmodeled dynamics in a system, it is not of key importance in this experiment. The main take away should be that as the gain of Motorlab and NERMLAB are increased, the resulting responses should closely follow the model's prediction, which in this case, would be a more oscillatory response without a change in the system's time constant. Table 6.3 tabulates the data collected from the Motorlab for this experiment.



(a) Theoretical step responses



(b) Experimental step responses



(c) Comparison plot for $K_p = 0.01$

Figure 6.6: Step responses that result from increasing proportional gain in a position control system for Motorlab

Table 6.3: Motorlab results from the conducted experiment compared with a theoretical model

| Gain ($\frac{V}{rad}$) | Theoretical CL Poles ($\frac{rad}{s}$) | Theoretical Period of Oscillations (<i>seconds</i>) | Measured Period of Oscillations (<i>seconds</i>) | Theoretical Time Constant (<i>seconds</i>) | Measured Time Constant (<i>seconds</i>) |
|------------------------------------------|------------------------------------------------------------------|------------------------------------------------------------------------|---------------------------------------------------------------------|---------------------------------------------------------------|------------------------------------------------------------|
| 0.0001 | $-1.2 \pm 4.6j$ | 1.37 | 1.32 | 0.86 | 0.86 |
| 0.001 | $-1.2 \pm 14.9j$ | 0.42 | 0.41 | 0.86 | 0.86 |
| 0.01 | $-1.2 \pm 47.1j$ | 0.13 | 0.13 | 0.86 | 0.86 |

Chapter 7

High Frequency Dynamics

Chapter 7 will cover the concept of 'high frequency dynamics'. High frequency dynamics in this context are the faster dynamics in comparison to the mechanical models in the Motorlab and NERMLAB systems. In chapter 7, the higher frequency, or faster dynamics, are a low pass filter on the output speed from the nominal plant¹.

The purpose of this experiment is to show that high frequency dynamics will effect controller design. In this context, if the gains of the system, specifically proportional gain, are made to be too large, the closed-loop system poles will run into the high frequency dynamics, invaliding the simplified model. The question is when is it valid to ignore these higher frequency dynamics. In Control of Mechanical Systems I at Kansas State University, this concept of ignoring system dynamics is handled using a rule of thumb:

"We can ignore open loop poles and zeros when they are more than 10 times larger (in terms of magnitude, which is the distance from the origin of the s plane) than the closed loop poles that result from ignoring them."

This experiment will demonstrate this rule of thumb by running multiple control systems with varying, increasing gains. It should demonstrate that, theoretically, when ignoring the higher frequency dynamics, one could potentially increase the gain of the system infinitely.

¹Nominal plant being the combination of the mechanical dynamics and derivative, as seen in figure 7.2

However, this does not play well on a real system. Care and due diligence should play a part in the design of a control solution. Pushing the gains too far out could result in system instability when the designer might not be aware of the higher frequency effects that are not included in a system or simplified dynamic model.

7.1 Mathematical Models

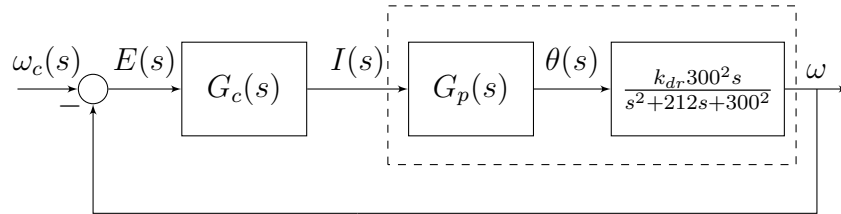


Figure 7.1: Closed Loop Speed Control System

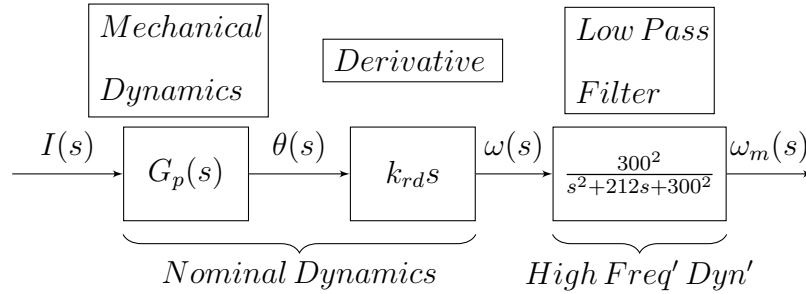


Figure 7.2: Open Loop System from Figure 7.1

Section 7.1 will develop the describing equations that are used as models for this experiment. Figure 7.1 shows a pictorial representation of the system that will be used in this experiment. Figure 7.2 is simply the another representation of the boxed section of figure 7.1, to emphasis the fact that the low-pass filter is filtering the noise that the derivative in this system produces.

The plant model for this experiment is simply a restatement of equation 4.20.

$$G = \frac{k_t}{RJ s + (Rb + K_E k_t)} \quad (7.1)$$

Additionally, since high frequency dynamics in this system exist (the low-pass filter), a transfer function of a low-pass filter (eq. 4.21) and derivative are needed:

$$G_{hf} = \frac{300^2 s}{s^2 + 212s + 300^2} \quad (7.2)$$

With Equations 7.1 and 7.2, it is possible to derive Equation 7.3, the final open loop system, which is just a reduction of the blocks in series from Figure 7.2.

$$G_{ol} = \frac{K_p k_t 300^2 s}{(RJ s + Rb + K_E k_t)(s^2 + 212s + 300^2)} \quad (7.3)$$

From here it is possible to arrive at Equation 7.4, the closed loop transfer function of equation 7.3:

$$T_{hf} = \frac{300^2 K_p k_t s}{(RJ s + Rb + K_E k_t)(s^2 + 212s + 300^2) + 300^2 K_p k_t s} \quad (7.4)$$

Likewise, the closed loop transfer function to the simplify model, Equation 7.1, is:

$$T = \frac{K_p k_t}{RJ s + (Rb + K_E k_t) + K_p k_t} \quad (7.5)$$

Equations 7.4 and 7.5 will be the base models to compare experimental results to.

7.2 Experiment

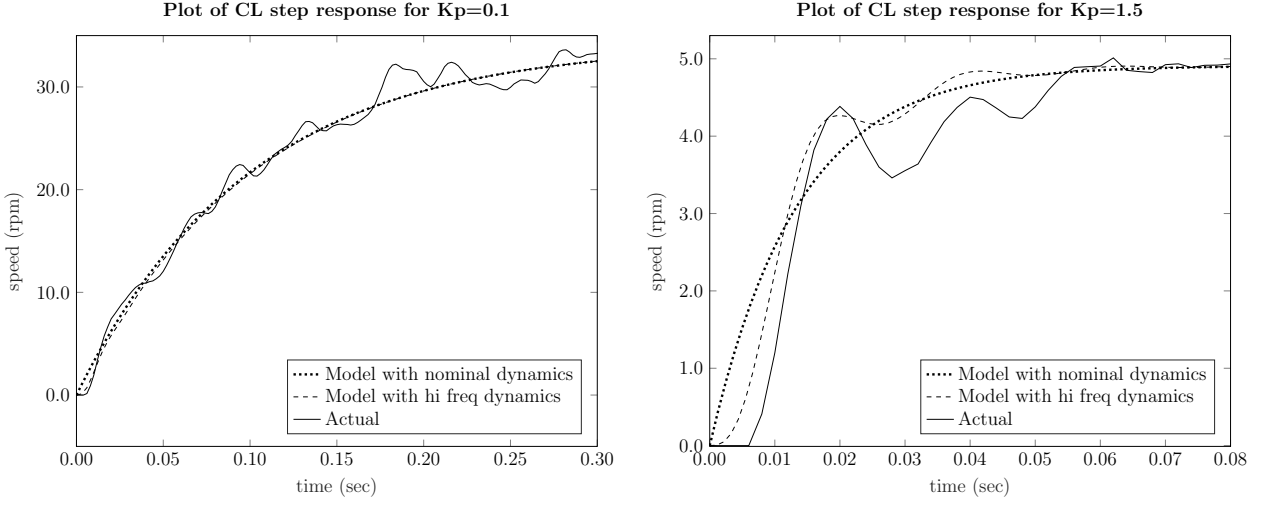
For this experiment, data will be collected for a series of four gains, listed in Table 7.1. Care has to be given when designing this experiment, because as you increase the proportional gain of the system, you start to command higher voltages. Since the NERMLAB saturates at 8V, it is easy to command more voltage than the motor driver can output, causing collected

data to not fit the models. In Table 7.1, the magnitude of the step is scaled with the gain to avoid exactly this issue. Other issues with this experiment will be the fact that the encoder does not have high accuracy, and results in undesired noisy data. This noisy data can further complicate the lab for students, as it can obfuscate the results. However, it will be shown that while the system is noisy, the higher frequency dynamics can be clearly seen from the results as the proportional gain is increased.

Table 7.1: Table of experimental gains to be used on the NERMLAB system

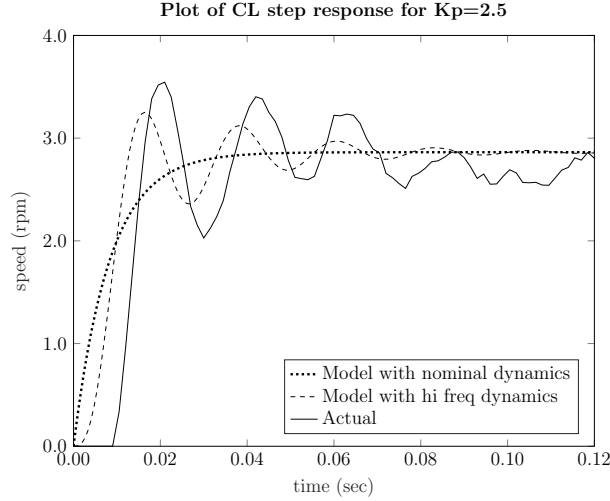
| Gain (V/(rad/s)) | Magnitude of Square Wave (rad/s) |
|-------------------------|-------------------------------------------------|
| 0.1 | 75 |
| 1.5 | 5.3 |
| 2.5 | 3.0 |
| 5 | 1.5 |

The NERMLAB is placed in speed control mode for this experiment, and a sample rate of 500 Hz is used to ensure the system is given enough time to get to steady state. For the final gain of 5, a special procedure is conducted to show the system as it grows to instability. It should demonstrate that pushing the gain out to far will result in an unstable system because of the higher frequency dynamics, which push the closed loop poles into the right hand side of the s-plane. This final procedure demonstrates the concept of knowing when it is ok to neglect certain dynamics in a simplified model.



(a) Closed Loop Step Response for $K_p = 0.1$

(b) Closed Loop Step Response for $K_p = 1.5$



(c) Closed Loop Step Response for $K_p = 2.5$

Figure 7.3: Step responses that result from increasing proportional gain in a speed control system for NERMLAB. A comparison between the high frequency model, simplified model, and the actual system response are show in these three figures.

Figure 7.3 shows the step responses for the first set of three gains. It can be seen that for $K_p = 0.1$, that noise is on the output response, however, the system does not show any signs of having higher frequency dynamics superimposed on the system, like the model predicts. For the second gain of 1.5 in Figure 7.3, the system starts to run into the higher frequency dynamics of the low pass filter. The model predicts superimposed oscillations on a first order

system. The NERMLAB's response seems to show that there is higher frequency effects in the system, although the noise from the encoder accuracy doesn't track the model as nicely as the Motorlab, which will be seen in the next section.

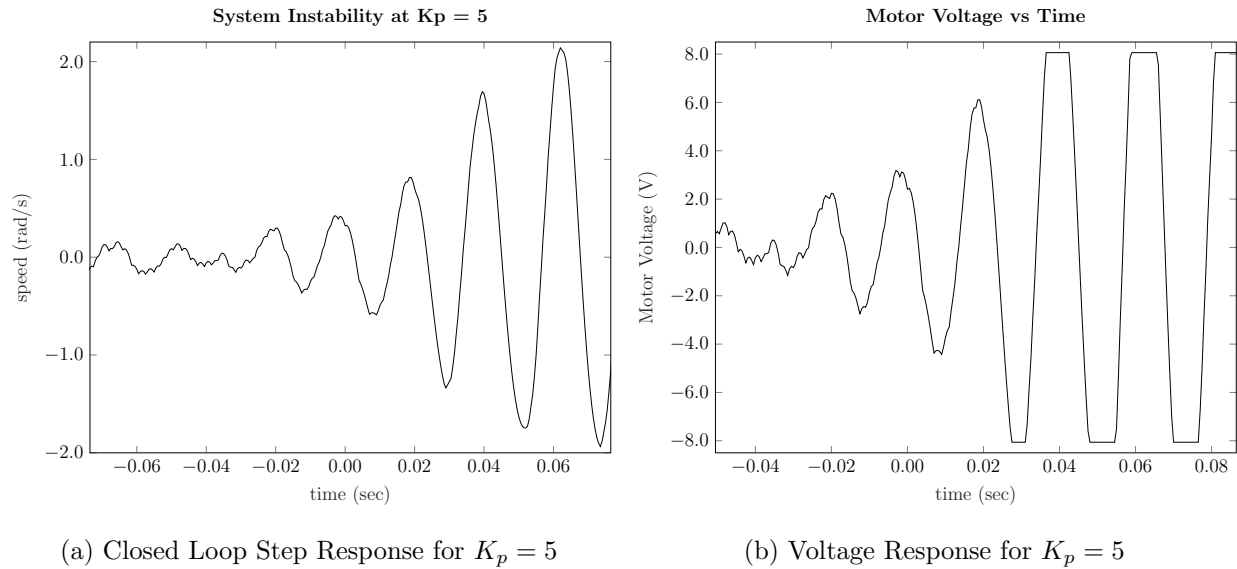
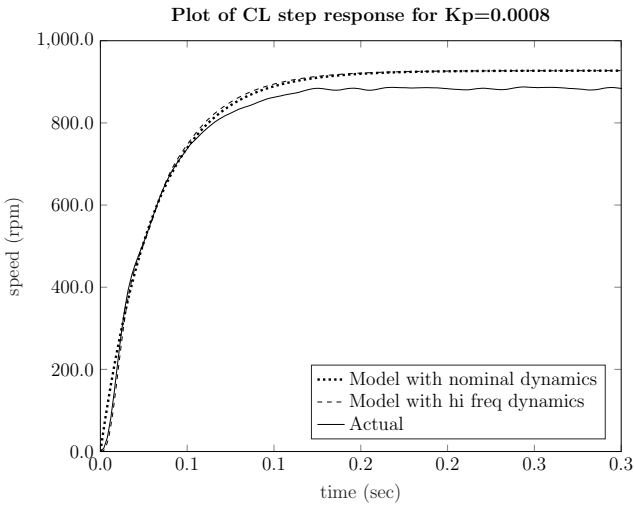


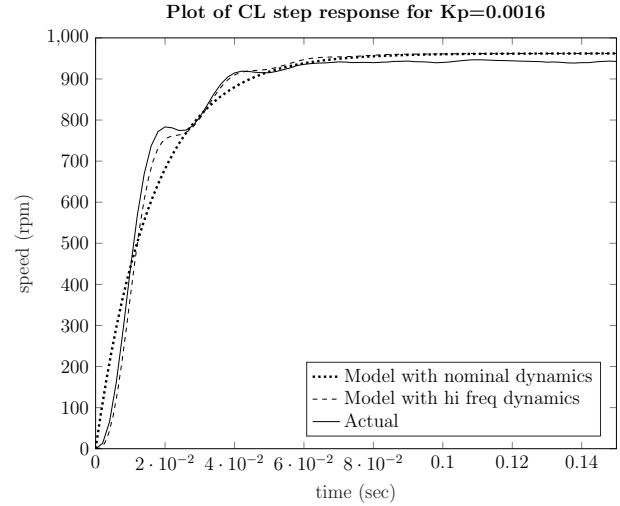
Figure 7.4: Step response shows how the NERMLAB system goes unstable when the proportional gain is increased too far.

As for the unstable gain of five, the system grows to instability as the model predicts (Figure 7.4a). In Figure 7.4b, the concept of a limit cycle is introduced, as the commanded voltage to the motor saturates at $\pm 8V$ and continues to cycle at that saturated voltage. Theoretically, the voltage command should continue to grow grow infinitely, so this concept of a limit cycle proves useful in instructing students about the physical limitations of a real system.

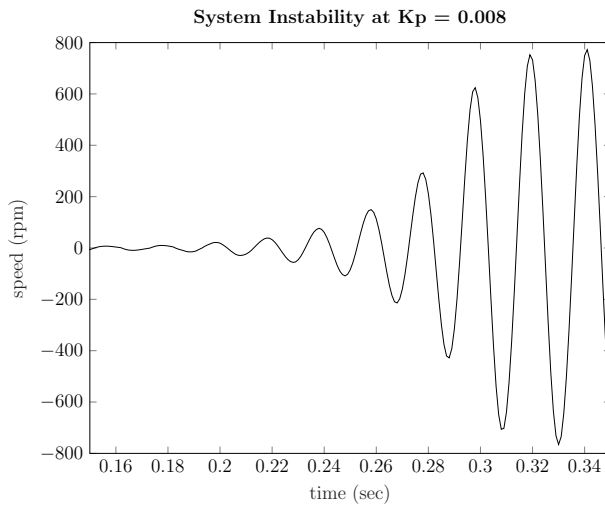
7.3 Motorlab Results



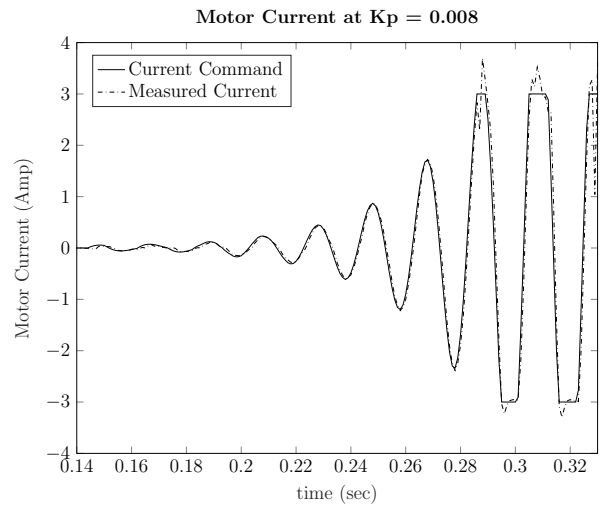
(a) Closed Loop Step Response for $K_p = 0.0008$



(b) Closed Loop Step Response for $K_p = 0.0016$



(c) Closed Loop Speed Response for $K_p = 0.008$



(d) Current Response for $K_p = 0.008$

Figure 7.5: Step responses that result from increasing proportional gain in a speed control system for Motorlab. A comparison between the high frequency model, simplified model, and the actual system response are shown in these two figures.

The biggest difference between the NERMLAB and Motorlab for this experiment were the encoder inaccuracies on the NERMLAB with speed control. This noise can cause students issues in conducting the experiment if they are not formally aware the effect it has on the

results. The experiment has to be carefully designed so that the higher frequency dynamics are clearly seen above the noise of the sensor. The Motorlab does a better job of tracking the actual model when comparing against the NERMLAB. However, as stated previously, the point of this exercise is to demonstrate the differences between a simplified model and the actual. Both systems do a good job of showing when higher frequency dynamics need to be included in a model.

Table 7.2: Table of experimental gains to be used on the Motorlab system

| Gain (A/RPM) | Magnitude of Square Wave (RPM) |
|---------------------|-----------------------------------------------|
| 0.0008 | 1000 |
| 0.0016 | 1000 |
| 0.008 | 50 |

The Motorlab had to use different gains and commanded values than the NERMLAB (Table 7.2), and is due to the Motorlab running current control. Additionally, the Motorlab could keep a constant magnitude in the command, as the NERMLAB suffered from saturation as the gain of the system increased, and this is a result of the Motorlab having larger operating limits. The final gain in Table 7.2 is used to show system instability at larger gains, which can be seen in Figure 7.5c.

One important thing to note when conducting this experiment is steady state error, which is the difference between commanded step size and the actual steady state response. It should be pointed out that the responses in Figures 7.3, never reach the commanded values in Table 7.1. This concept will be later explored in this thesis, but does not influence the results presented in this chapter.

Chapter 8

Frequency Response of a Position Control System

Chapter 8 introduces the topic of frequency response of a position control system. This experiment will follow the procedure outlined in laboratory 10 (Appendix C), with some additions. The Motorlab apparatus that is used at Kansas State University utilizes a spring coupled to the BLDC motor's inertia to demonstrate to students the idea of a frequency response of an actual system. The results are then compared to a developed model. Typically, the Motorlab apparatus follows the developed model quite accurately; however, with this laboratory, an improved model is generated to show more clearly what the real system is doing and how the real system can vary slightly from the theoretical. Students are also introduced to the concept of resonance and how to approximate it from the natural frequency (ω_n). The setup for the NERMLAB is different, and in the NERMLAB's case, a second order system is generated by looking at a closed loop position control loop. Students are to collect data at five discrete points that characterize the overall system's response at varying frequencies. In the case of both the NERMLAB and Motorlab, this second order system has a resonant peak and a $40 \frac{dB}{decade}$ drop off at higher frequencies. In order to completely characterize the frequency response, the natural frequency is used as a center frequency in

a range of $\frac{\omega_n}{10}$ to $2\omega_n$. This frequency range is used to capture the important features of the frequency response for both systems.

8.1 Mathematical Model

Just like in chapter 6, a closed loop position control system is used and the formulation is nearly the same, with one addition. In the case of this position control system, a static gain of 20 is used, and must be included in the model development. Looking at equation 6.3, K_p is the only value that needs to be substituted to complete the model. Substituting K_p into equation 6.3 gives the result:

$$T = \frac{20k_t}{RJs^2 + (Rb + K_E k_t)s + 20k_t} \quad (8.1)$$

8.2 Experiment

Four experiments were conducted to get an average value for the NERMLAB. It was found through experimentation, by driving the NERMLAB at slightly smaller values than the calculated natural frequency, that the NERMLAB has a resonant frequency of 4.6 Hz. This frequency resulted in the nearest 90 degree phase shift, as seen in Figure 8.3, and largest amplitude ratio between the commanded position and the actual output of the system. The resonant frequency is slightly smaller than the predicted frequency of the model; however, this new calculated value will be used to generate an improved frequency response of the NERMLAB that better describes the actual system.

Table 8.1 hosts the averaged values for the four experiments conducted on the NERMLAB. Each experiment varied the frequency slightly to get a broader range of data points for the frequency response. Each of the averaged values for the all of the frequencies are plotted in Figure 8.1, along with the improved model that the collected data followed.

Table 8.1: Experimental Results for Position Control Frequency Response

| Freq' (Hz) | $\frac{\omega_n}{10}$ | $0.75\omega_n$ | ω_n | $1.25\omega_n$ | $2\omega_n$ |
|--------------------------|-----------------------|----------------|------------|----------------|-------------|
| Input (rad) | 0.32 | 0.32 | 0.22 | 0.32 | 0.32 |
| Freq' (Hz) | 2.80 | 21.67 | 28.90 | 36.14 | 57.81 |
| Mag' (dB) | -0.66 | 3.75 | 9.00 | 4.00 | -6.00 |
| Phase Shift (deg) | 2.5 | -27.57 | -84.10 | -141.52 | -173.92 |

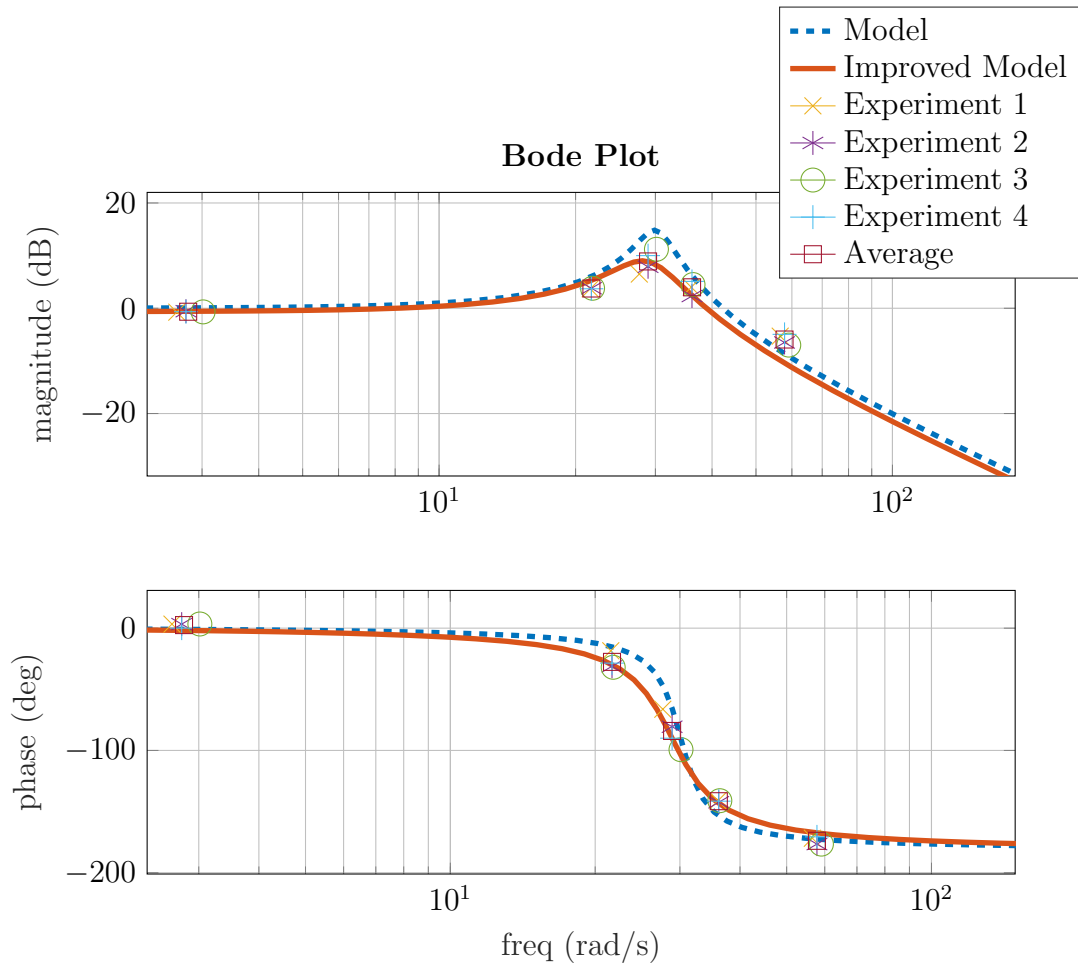


Figure 8.1: Bode plot of a position control system. A model and an improved model, along with collected data, and averaged data, can be seen in this figure.

As expected, there were some fluctuations in the output phase and magnitudes of each conducted experiment; however, both the averaged data and all four experiments seem to follow both the improved and actual model well. The model predicted that at frequencies lower than that of the natural frequency, the output phase and magnitude should track relatively well, meaning the phase shift was zero degrees and the magnitude ratio is one. At the natural frequency, a larger output amplitude, along with a 90 degree phase shift, should be observed. At frequencies greater than that of the natural frequency, attenuation should occur and the phase shift should be approaching 180 degrees as the frequency is increased further. By all accounts, the NERMLAB demonstrated these observed model trends adequately. The main difference between the two models is that the actual model predicts a larger resonant peak at the resonant frequency. Comparing the two magnitudes, the actual system predicts a $14.8dB$ peak, while the system presented a lower $9dB$ (averaged) peak value, which could be a result of static friction.

$$\frac{\frac{20k_T}{RJ}}{s^2 + \frac{(Rb+K_e k_T)}{RJ}s + \frac{20k_T}{RJ}} = \frac{K_{DC}\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (8.2)$$

The above experiment that developed a frequency response for the NERMLAB can be done by following the listed steps:

1. Calculate the natural frequency by comparing the developed model, Equation 8.1¹, to a standard second order system, and solving for ω_n , as seen in Equation 8.2.

$$\omega_n = \sqrt{\frac{20k_T}{RJ}} = 30.3 \frac{rad}{s} \text{ or } 4.82Hz$$

2. Use the calculated ω_n that was found in step one to experimentally determine the resonance of the NERMLAB by driving the system around this frequency, varying the input frequency, until 90 degrees of phase shift occur. A smaller command signal

¹This equation is in standard form

should be used when approaching the resonant frequency to minimize damage to the system.

3. Once the resonance is known, drive the system at the frequencies tabulated in row one of Table 8.1. From here, it is necessary to then use Equations 8.3 - 8.5 to find the phase lag and magnitude ratio of the output. Figure 8.3 shows how the data for the timelag at the resonant frequency was gathered to calculate the phase shift and magnitude ratio. Figure 8.2 shows what the values in Figure 8.3 represent when using Equation 8.3. The same process is carried out for the rest of the frequencies.

$$t_{lag} = \frac{|t_{peak} - t_{valley}|}{2} - t_{crossing} \quad (8.3)$$

$$\phi = 360t_{lag}f_c \quad (8.4)$$

$$Mag' Rat' = \frac{y_{max}^{Actual} - y_{min}^{Actual}}{y_{max}^{Command} - y_{min}^{Command}} \quad (8.5)$$

4. Generate an improved model (G_{new}) from the data. The new model can be developed by using equation 8.6. Here, K_{DC} is found by looking at the magnitude ratio at $\frac{1}{10}$ the natural frequency. Likewise, ζ can be calculated by realizing that at the natural frequency ($s = j\omega_n$), the magnitude ratio is $M_{\omega_n} = \frac{K_{DC}}{2\zeta}$. Here ζ is simply $\frac{K_{DC}}{2M_{\omega_n}}$.

$$G_{new} = \frac{K_{DC}\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (8.6)$$

Note that Equation 8.3 is the time lag of waveforms. It is the mean of the time values of the peak and valley of the output waveform. Also $t_{crossing}$ is simply when the time of the input crosses 0 degrees. The magnitude ratio is calculated by finding the max and min values of the output and input and dividing respectively.

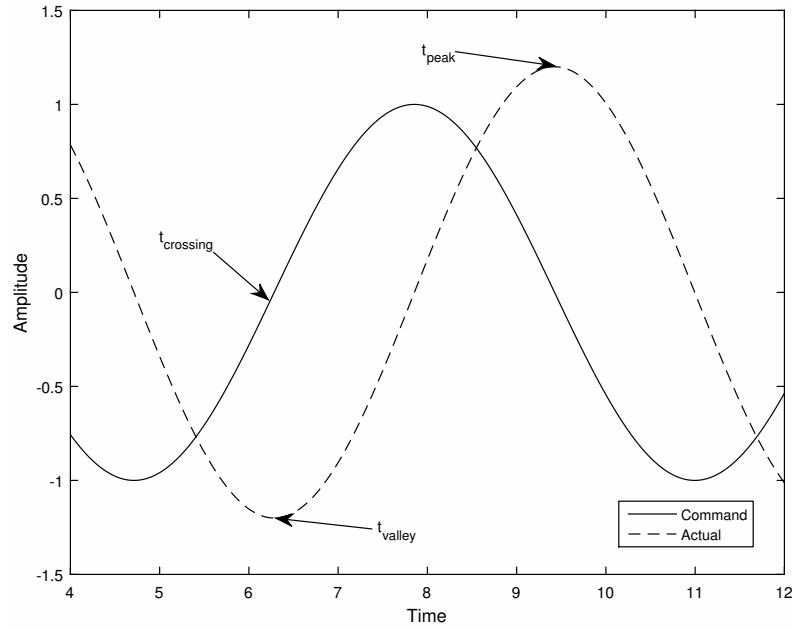


Figure 8.2: The phase lag can be calculated by finding the time difference between the two waveforms using the annotated positions in the figure.

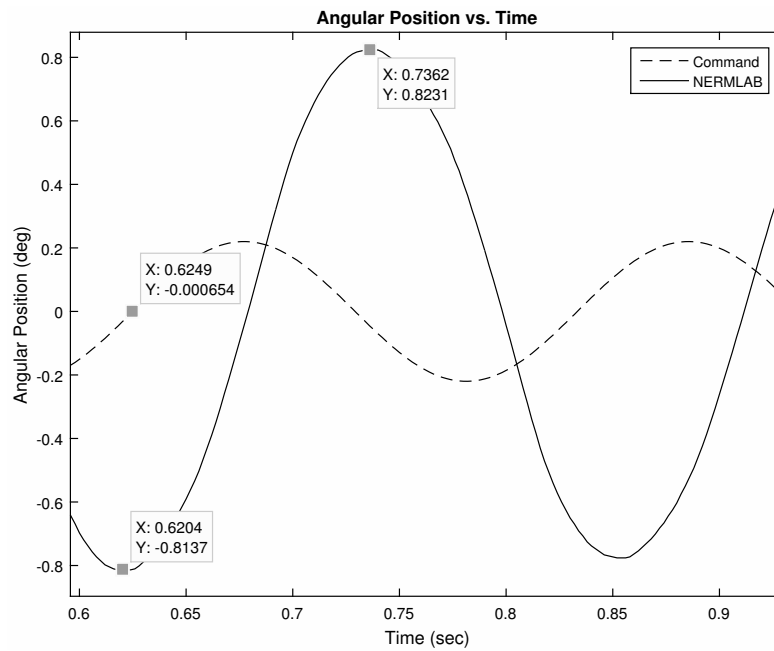


Figure 8.3: NERMLAB's 90 degree phase shift at the resonant frequency.

New Parameter Estimation

From the improved frequency response, new system parameters can be estimated by comparing the original starting model to that of a second order system, as seen in Equation 8.2. Here J, R, K_E are assumed to be known values, as they can be directly measured and approximated. From Equation 8.2, the following equations can be derived:

$$\begin{cases} K_t = \frac{RJK_{DC}\omega_n^2}{20} \\ b = \frac{2RJ\zeta\omega_n - RK_E K_t}{R} \end{cases} \quad (8.7)$$

Here, K_t for the improved model is found to be $0.009 \frac{N \cdot m}{A}$, giving 10 percent relative difference from the actual value of K_t . Likewise, a new friction estimate can be calculated using Equations 8.7. b was found to be 4.87×10^{-4} , leading to a relative difference of 63 percent. However, when the experiment was conducted on the Motorlab, the new estimate for the friction value also led to a similar relative difference of 63 percent. This can be accounted for by the fact that the linear model has non-linear effects that are not accounted for when the original estimate for friction was made. The relative difference for friction could also be high due to the fact the improved model was generated from averaged values. For example, ζ depends on the magnitude ratio at the resonant frequency. If the largest magnitude of all four experiments conducted is used, which in this case is experiment three, the relative difference of friction drops to 23 percent. These new estimates depend heavily upon how accurate the results from the experiment are tabulated and collected.

8.3 Motorlab Results

While the two frequency response experiments differed in the way they were conducted, the end objective of developing a frequency response on both systems was a success. Both systems have the ability to translate the idea of resonance and generating improved models based on collected data. One potential drawback of using the NERMLAB over the Motorlab

is that the resonant frequency is not quite as obvious. When doing this experiment on the Motorlab, once the resonant frequency is found, the system provides audible feedback via the buzzing and humming of the spring, where as the NERMLAB, this audible feedback is not present. In the NERMLAB's case, due to the lack of additional experimental feedback mechanisms, has to have the output plotted with each trial to ensure the system is getting near 90 degrees of phase shift. Figure 8.4 shows the experiment conducted on the Motorlab. Just like the NERMLAB, the data follows the improved model for the Motorlab accurately. Table 8.2 shows the collected data for the Motorlab experiment.

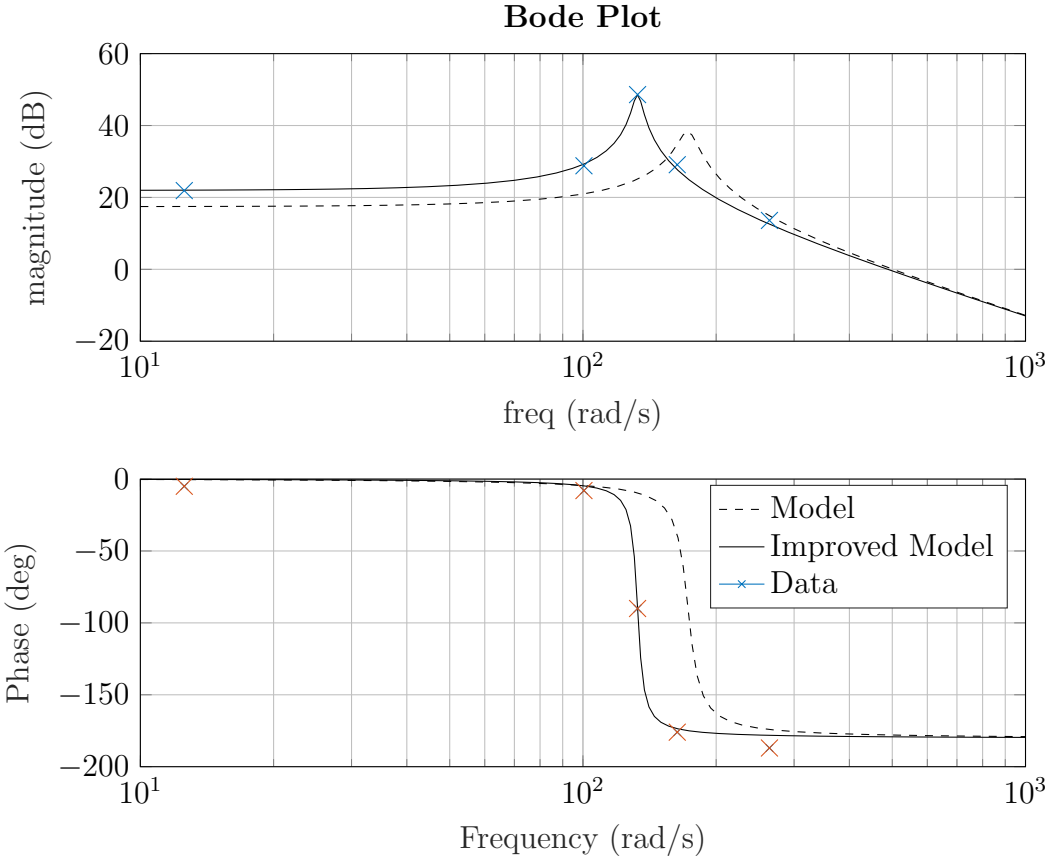


Figure 8.4: Bode Plot of a Second Order Spring System

Table 8.2: Experimental Results for Position Control Frequency Response on Motorlab

| Freq' (Hz) | $\frac{\omega_n}{10}$ | $0.75\omega_n$ | ω_n | $1.25\omega_n$ | $2\omega_n$ |
|------------------------------|-----------------------|----------------|------------|----------------|-------------|
| Input (Amp) | 1 | 1 | 0.25 | 1 | 2 |
| Freq' (Hz) | 2 | 16 | 21.15 | 26 | 42 |
| Mag' (dB) | 21.9 | 28.8 | 48.6 | 29.1 | 13.6 |
| Phase Shift (deg) | -5 | -8 | -90 | -176 | -187 |

Equations 8.9 can be used to calculate the new system parameters for the improved model for Motorlab. The previous section commented on these values, and the relative difference that is generated from the actual system parameters. The new system parameters are taken by comparing a second order system with the actual model, which in the case of the Motorlab, includes a spring.

$$G = \frac{k_{dr}K_T}{Js^2 + bs + K_s} \quad (8.8)$$

$$\left\{ \begin{array}{l} K_s = J\omega_n^2 = 0.2278 \\ b = 2\zeta\omega_n J = 8.0256e - 05 \\ K_t = \frac{K_{DC}K_s}{k_{dr}} = 0.0497 \end{array} \right. \quad (8.9)$$

Actual values for the Motorlab system parameters can be found in Appendix E.

Chapter 9

Effects of Integral Control on Steady State Error

Chapter 9 introduces the concept of system type and its effect on steady state error. Steady state error is the error between a commanded value and the value of the actual steady state response. Where as system type is defined as the number of free integrators of the open loop transfer function, which occur when one or more of the poles of the system is equal to zero, e.g. $s^n = 0$. System type changes based on input, and effects steady state error. For example, if a system has a free integrator and a unit step is generated, the response will have zero steady state error and will have a system type of one. Where as if a system is type one, but a ramp input is generated, the steady state error will be non-zero, meaning the response never makes it to the commanded value. Table 9.1 lists the stead state error based on system type and input. This table is used in Control of Mechanical Systems I at Kansas State University.

Table 9.1: Steady State Error

| Input | Type: | 0 | 1 | 2 |
|-----------|----------|----------|--------|--------|
| Step | Finite | 0 | 0 | 0 |
| Ramp | ∞ | Finite | 0 | 0 |
| Parabolic | ∞ | ∞ | Finite | Finite |

The idea of system type becomes clear when running speed controllers on both the NERMLAB and Motorlab systems. In the experimental portion of Chapter 9, it will be seen that when using strictly a proportional controller with a step input, a finite steady state error occurs because of the number of free integrators in the open loop transfer function (Equation 4.20). This effect was not observed in other experiments in this thesis involving a position controller, because the system type in the case of both the NERMLAB and Motorlab is one. The main point of this experiment will be to show that increasing the proportional gain of the system leads to improvement of the steady state error, but never minimizes it to a non-zero value. Later, it will be proven that by simply implementing a Proportional-Integral (PI) controller, the steady state error should minimize to zero, as Table 9.1 predicts. This is due to the fact that when adding integral action to a controller, a free integrator is added to the open loop system and this will be shown in section 9.1.

9.1 Mathematical Model

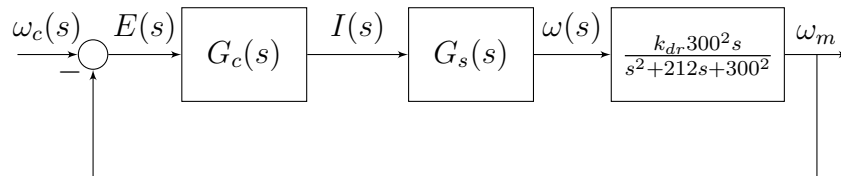


Figure 9.1: Closed Loop Speed Control System

In Figure 9.1, G_s is the plant speed model that was developed earlier in this thesis, and is restated below:

$$\frac{\omega(s)}{V(s)} = \frac{k_T}{RJ s + (Rb + K_e k_T)} \quad (9.1)$$

As has been the case for a majority of the experiments in this thesis, G_c , the controller, is simply equal to K_p . This controller will be used for the first portion of the experiment. However, since the purpose of this chapter is to show how to minimize steady state error, a controller involving integral action must be developed. An integrator in the s-domain is simply $\frac{1}{s}$, or in the case of the controller for this experiment, $\frac{K_i}{s}$, where K_i is the integral gain of the system. Adding the separate integral and proportional controllers together gives a PI controller, stated as equation 9.2.

$$G_c = K_p + \frac{K_i}{s} \quad (9.2)$$

However, it is more convenient to place Equation 9.2 in Zero-Pole-Gain (ZPK) form, as it is much easier to pull the exact location of the zero from the system in this form. Factoring Equation 9.2 results in the following equation:

$$G_c = \frac{K_p(s + \frac{K_i}{K_p})}{s} \quad (9.3)$$

Where the zero, z , is equal to $-\frac{K_i}{K_p}$.

$$G_c = \frac{K_p(s + z)}{s} \quad (9.4)$$

9.2 Experiment

Table 9.2 lists the various gains to be used in this experiment. The experiment starts with a speed controller commanding a step size of $50 \frac{rad}{s}$, while the proportional gain is increased

for a set of three gains: 0.1, 0.15, and 3. For the third row in Table 9.2, an integral gain, K_i is set to 1 to add a zero to the system at $-10 \frac{rad}{s}$, and likewise, add a free integrator. It will be shown in this experiment that by adding this free integrator, the steady state error will go to zero. Note that in Table 9.2, for the final gain of three, that the step size changes. This is due to the fact that the NERMLAB saturates quickly when using a $50 \frac{rad}{s}$ command. The gain of three is used to show that simply increasing the proportional gain does not get rid of the finite error in the system, but should improve it. It will also demonstrate that higher frequency dynamics, such as the low pass filter, exist in the system, and result in more oscillations of the response as the gain is increased further.

Table 9.2: Table of experimental gains to be used on the NERMLAB system

| K_p | K_i | Magnitude of Step (rad/s) |
|-------|-------|------------------------------|
| 0.1 | 0 | 50 |
| 0.15 | 0 | 50 |
| 0.1 | 1 | 50 |
| 3 | 0 | 2.6 |

Figure 9.2 shows the experimental results for the NERMLAB. In Figure 9.2a, as the proportional gain was increased, the error between the commanded value and the actual steady state response decreased; however, the system's error never reduced to zero. Once the PI controller was added, the steady state error became zero. The benefits are PI control in this experiment are evident, as the proportional gain of the system can be reduced, meaning the tracking of the system improves, fewer oscillations are produced, and the system achieves a similar settling time as that of an increased proportional gain. Figures 9.2a - 9.2d show how the NERMLAB tracked the individual models for each set of gains.

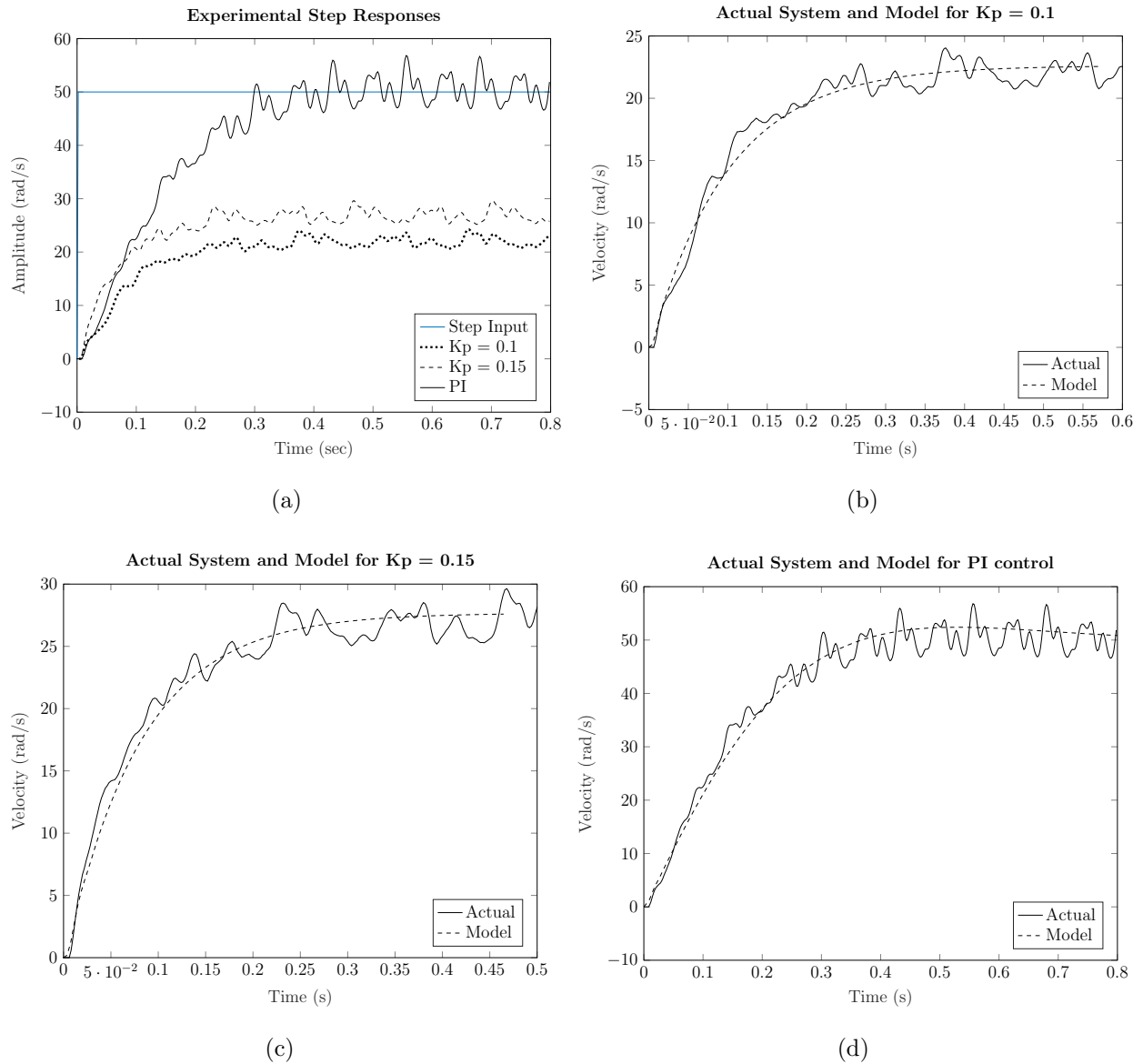


Figure 9.2: NERMLAB's steady state error, showing how increasing the proportional gain does not decrease the error until PI control is used.

What can be seen in Figure 9.3 is the system's closed loop poles running into the dynamics of the low pass filter, causing oscillations, which would continue to grow with an increasing gain. While the error was reduced even further than that of the lower proportional gains, the model still predicts a finite error between the command and response. The finite error is harder to see from the actual system results because of the inaccuracy of the encoder,

but concept is still effectively demonstrated when looking at the collected results in Table 9.3.

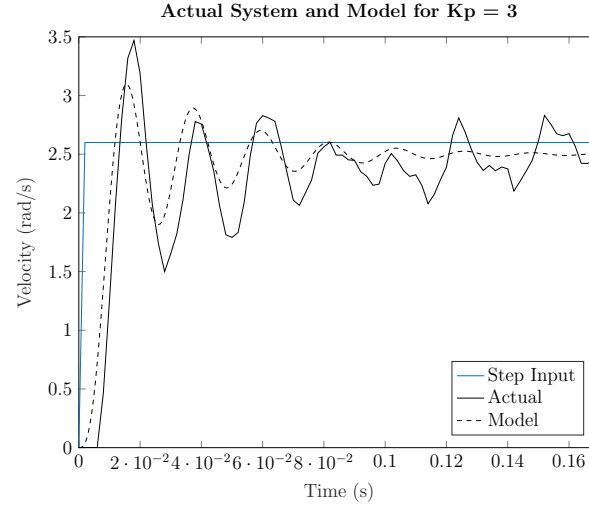


Figure 9.3: Steady state error at a higher proportional gain of $K_p = 3$.

Table 9.3: Collected data from theoretical model and actual system for the NERMLAB

| Gains | | Model | | | | Experiment |
|-------|-------|----------|------------------|------------------------------------------------|-------|------------------|
| K_p | K_i | K_{DC} | SS Speed (rad/s) | Poles | Zeros | SS Speed (rad/s) |
| 0.1 | 0 | 0.45 | 22.62 | $-103.7 \pm 279.8j,$ -10.1 | 0 | 21.88 |
| 0.15 | 0 | 0.55 | 27.67 | $-102.5 \pm 279.4j,$ -12.5 | 0 | 26.75 |
| 0.1 | 1 | 1 | 50 | $-103.7 \pm 279.8j,$ $-10.1, -5.0 \pm 4.5j$ | -10 | 49.92 |
| 3 | 0 | 0.96 | 2.50 | $-30.7 \pm 283.9j,$ -156.2 | 0 | 2.44 |

Table 9.3 shows the collected data for the set of four gains. For the experimental steady state speed seen in column seven, a mean had to be taken for the final seconds of the data because of the noisy encoder. The NERMLAB follows the model adequately, as the difference

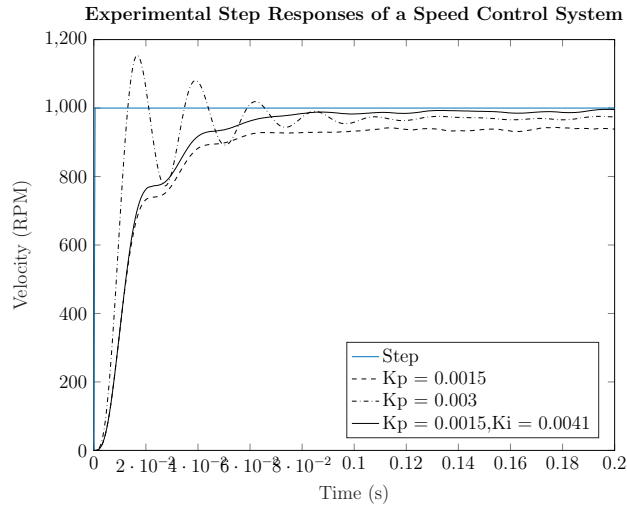
between the predicted steady state speed of the model and the actual system is minimal. Table 9.4 shows the steady state error of both the model and NERMLAB for the set of four gains.

Table 9.4: Steady state error for each set of gains

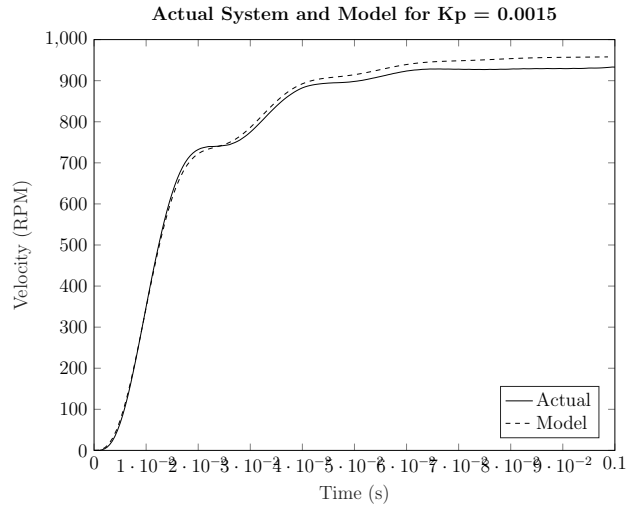
| K_p | K_i | Steady State Error | Steady State Error |
|-------|-------|--------------------|--------------------|
| | | (Model) | (Experiment) |
| 0.1 | 0 | 27.38 | 28.12 |
| 0.15 | 0 | 22.33 | 23.25 |
| 0.1 | 1 | 0 | 0.08 |
| 3 | 0 | 0.1 | 0.16 |

9.3 Motorlab Results

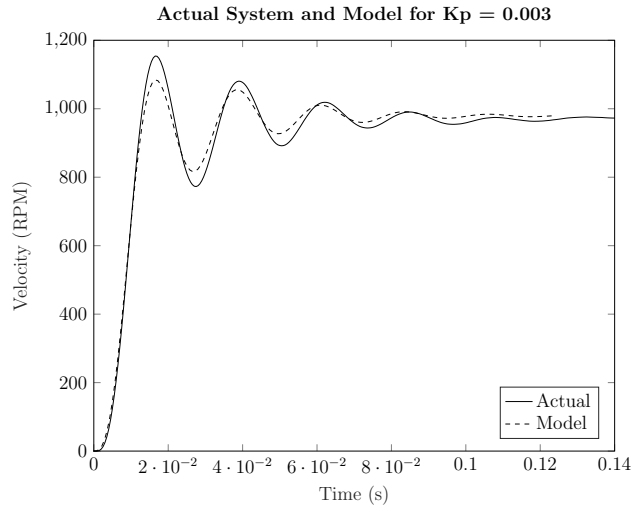
Both the NERMLAB and Motorlab demonstrate the concept of steady state error and system type comparably. Figure 9.4 shows the results from the experiment carried out on the Motorlab. It is evident that the Motorlab's tracking of the model is more effective because of the NERMLAB's encoder noise. However, the NERMLAB's results are more visual, meaning the effects of steady state error are more apparent when only proportional gain is used (Figure 9.2a).



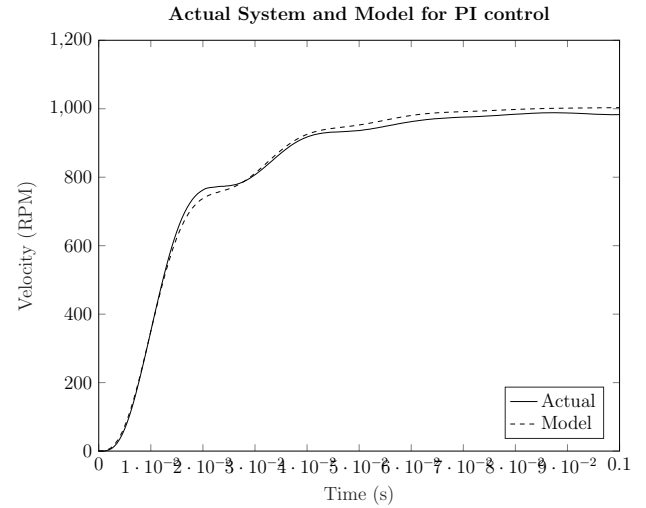
(a)



(b)



(c)



(d)

Figure 9.4: Motorlab's steady state error, showing how increasing the proportional gain does not decrease the error until PI control is used.

Table 9.5 lists the results found when the same experiment was conducted on the Motorlab. Note that because the Motorlab uses a current controller, the selected gains used are different from that of the NERMLAB.

Table 9.5: Collected data from theoretical model and actual system for the NERMLAB

| Gains | | Model | | | | Experiment |
|--------|---------|----------|------------------|---------------------------------|-------|------------------|
| K_p | K_i | K_{DC} | SS Speed (rad/s) | Poles | Zeros | SS Speed (rad/s) |
| 0.0015 | 0 | 0.960 | 960 | $-75 \pm 274j,$ -64 | 0 | 943 |
| 0.003 | 0 | 0.979 | 979 | $-43 \pm 278j,$ -128 | 0 | 976 |
| 0.0015 | 0.00405 | 1 | 1000 | $-75 \pm 274j,$ $-62, -2.72$ | -2.7 | 1000 |

Chapter 10

Derivative Action on a Position Control System

Derivative control, or more specifically Proportional-Derivative (PD) control, is a powerful tool in control design. However, one problem with derivative action in a controller is that it saturates the controller output in physical systems. As a result, it is difficult to obtain step responses that match their theoretical counterparts. Because of the saturation the derivative creates, the energy of the model will be greater than the energy the system actually receives. One way to combat this problem would be to use small command sizes. However, it is much easier to capture the effects of a linear model on the actual system using a ramp or triangle wave input.

A derivative controller adds a zero to the open loop system, as will be seen in the next section. For the NERMLAB and Motorlab system, this zero acts as a magnet, dragging the closed loop poles towards the left hand side of the s-plane, as seen in Figure 10.1. It was shown throughout this thesis that when using strictly a proportional controller, the system only becomes more oscillatory as the gain is increased. However, with a PD controller, the system remains oscillatory for a brief period before the closed loop poles merge onto the real axis as the gain is increased further. One pole of the system is captured by the zero,

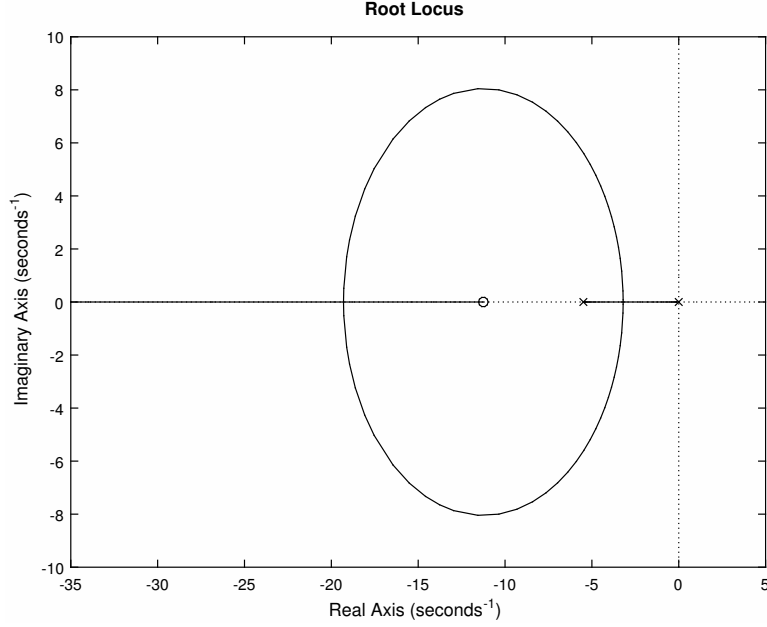


Figure 10.1: Root locus of the open loop plant and PD controller. Here the 'x' are the poles and 'o' the zeros.

while the other travels off further into the left hand plane, decreasing the settling time. This makes PD useful as a controller because the system can be made faster by moving the pole further out, i.e. increasing the gain of the system.

10.1 Mathematical Model Development

The same position model plant that has been used throughout this thesis will be used in this experiment, it is restated below:

$$\frac{\theta(s)}{V(s)} = \frac{k_t}{RJs^2 + (Rb + K_E k_t)s} \quad (10.1)$$

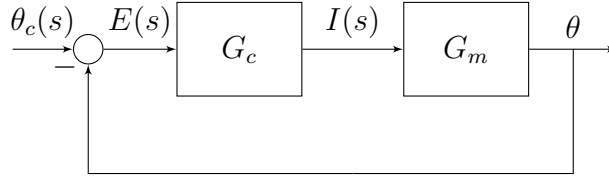


Figure 10.2: Closed Loop Control System

G_c changes in this experiment to a PD controller, meaning a derivative gain (K_d) is added with a proportional gain, as in Equation 10.2.

$$G_c = K_p + K_d s \quad (10.2)$$

It is possible to simplify Equation 10.2 further, reducing it to more convenient ZPK format.

$$G_c = K_d \left(s + \frac{K_p}{K_d} \right) = K_d (s + z) \quad (10.3)$$

In Equation 10.3, the ' z ' is the zero of the system, which is simply equal to $\frac{K_p}{K_d}$, or in the case of this experiment, $15 \frac{\text{rad}}{\text{s}}$.

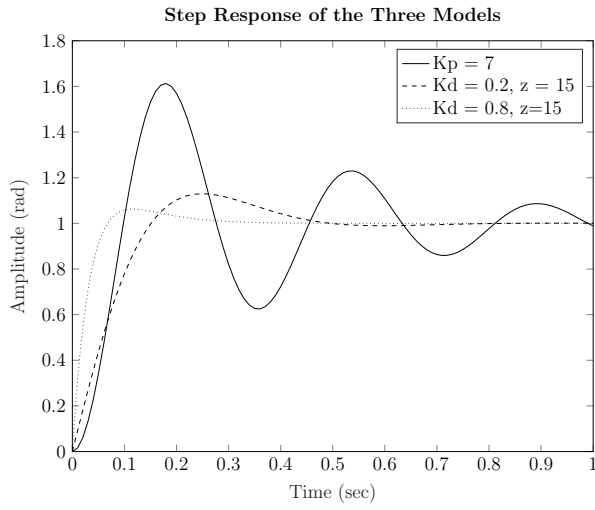
10.2 Experiment

Table 10.1: List of gains to be used in PD control experiment

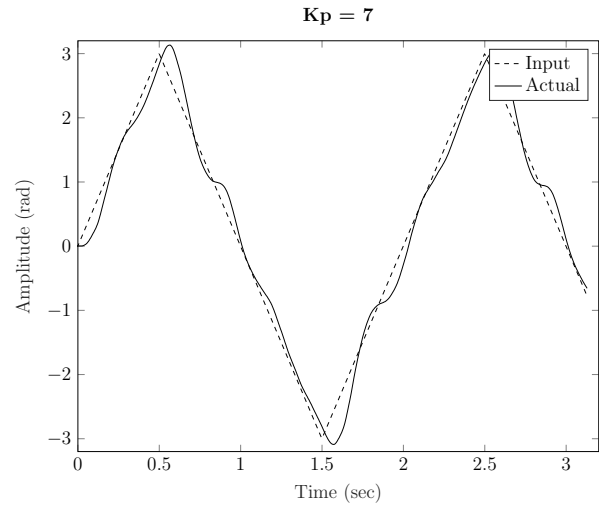
| K_p | K_d | Magnitude of Ramp (rad) |
|-------|-------|----------------------------|
| 7.0 | 0.0 | 3.0 |
| 3.0 | 0.2 | 3.0 |
| 9.0 | 0.8 | 3.0 |

Table 10.1 lists the gains that will be used in this experiment. For the triangle wave input, a magnitude of 3 *rads* will be used for all controllers. Likewise, a wave frequency and a sample frequency of 0.5 *Hz* and 303 *Hz*, respectively, will be used to generate a triangle wave with one full period.

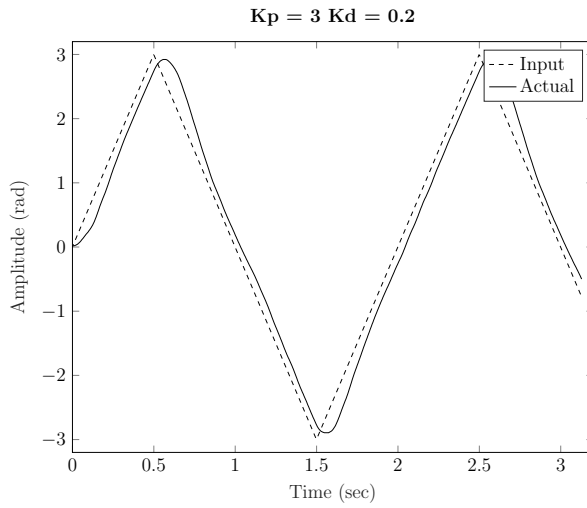
Figure 10.3 shows the experimental results, along with the theoretical step responses for the set of three gains. While the the results from the experiment can not be directly compared to the step responses of the theoretical model, things like the period of oscillation can be predicted. For the first gain of seven, the model indicates that the oscillation period should be 0.35 seconds, where as the actual response results in a oscillation period of 0.32 seconds. This result can be found by looking at where the waveform of the actual response crosses the command signal, as in Figure 10.4. Then, it can be compared with the calculated period of oscillation (0.35 seconds), $\frac{2\pi}{\omega_d}$, by looking at the imaginary portion of the poles of the closed loop transfer function.



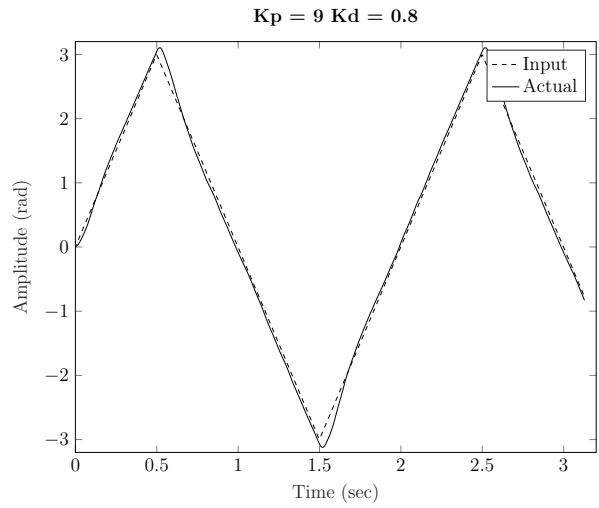
(a)



(b)



(c)



(d)

Figure 10.3: NERMLAB PD Controller Experimental Results.

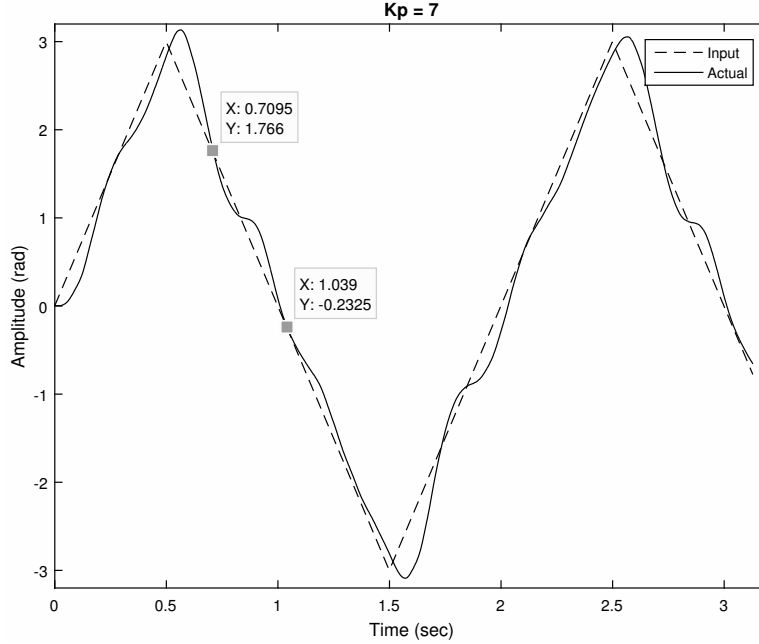


Figure 10.4: Period of Oscillation Calculation

One thing the model does not predict in this experiment is the steady state error. Recall the fact, that from Table 9.1, a step input for a type one system will produce zero error. Equation 10.1 has exactly one free integrator, and as a result, zero steady state error. However, for this experiment a ramp input is being used, which predicts a finite error that can not be seen in Figure 10.3a, but clearly visible in Figure 10.3c. Theoretically, by increasing the gain of the system, a reduction of this steady state error should occur. This can be easily verified by looking at a plot of the angular error vs time, as in Figure 10.5. With the inclusion of the higher gain, K_d , the magnitude of the error decreases from 0.24 *rad* to 0.08 *rad*, from the set of two K_d gains. The reduction in steady state error predicts better tracking for the third system and is a result of increasing the low frequency gain in the frequency response, as seen in Figure 10.6. Figure 10.3d demonstrates how simply increasing the gain of the system reduces steady state error, and in return, improves the tracking ability of the system.

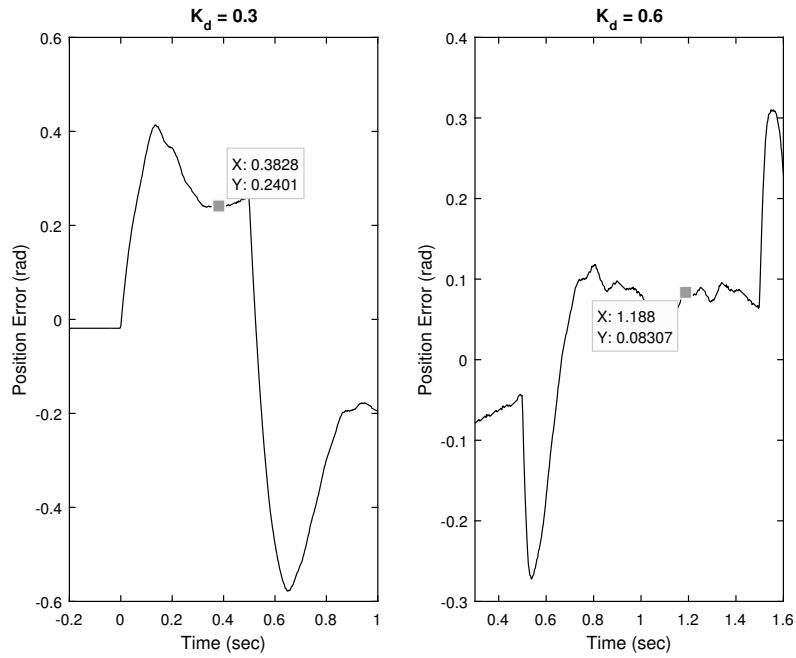


Figure 10.5: Angle Error that results in using PD control on NERMLAB.

Bode Diagram

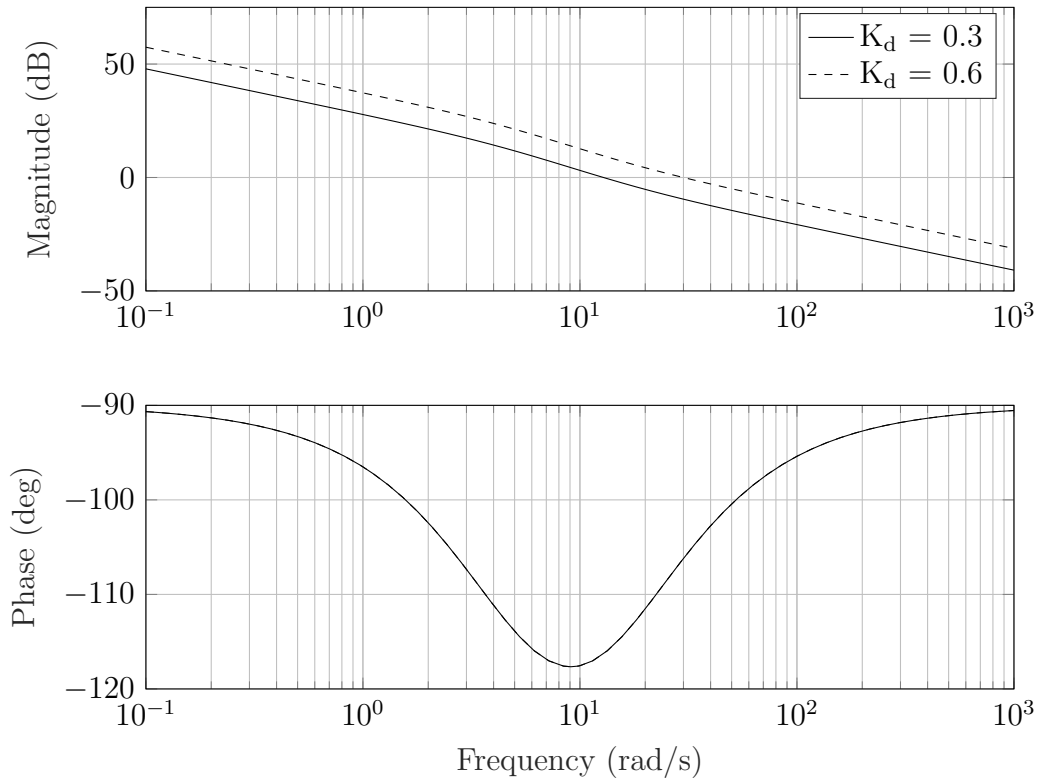


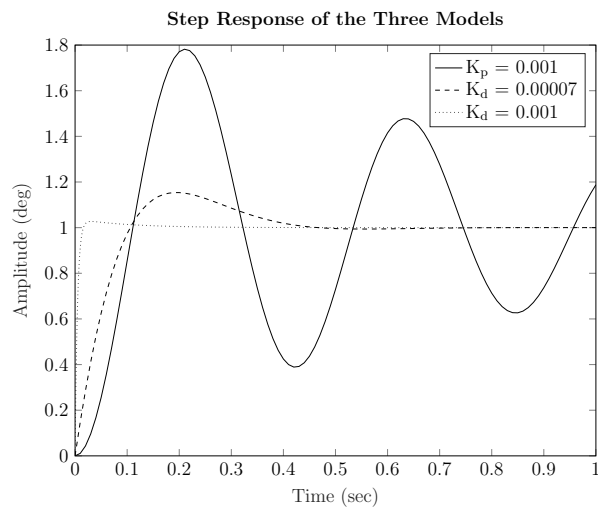
Figure 10.6: Steady state error at a higher proportional gain of $K_p = 3$.

10.3 Motorlab Results

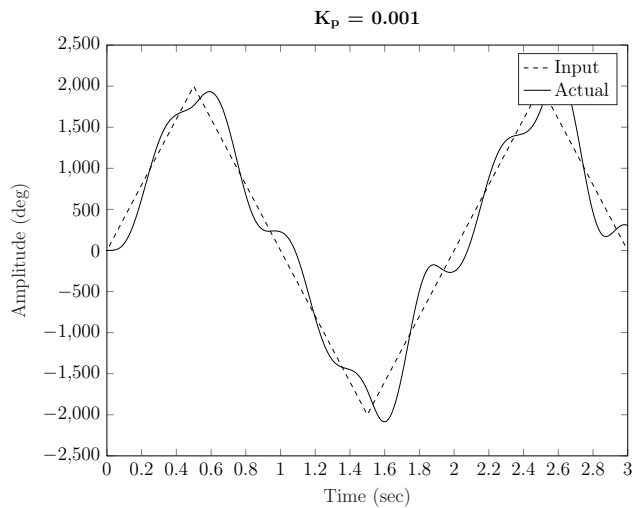
Figure 10.7 shows the results from the experiment conducted on the Motorlab. The procedure was exactly the same as conducted in the experiment portion of this chapter. Since Motorlab uses a current controller, a different set of gains are used, which are tabulated in Table 10.2. The Motorlab does a good job of demonstrating to students the effect derivative action has on a system's response. Unlike the NERMLAB, the Motorlab has higher operating limits, which allows the gains to be increased further, allowing for quicker responses, like of that depicted in Figure 10.7d. The NERMLAB had issues with saturation when the gains were increased to far, which limits the amount of derivative action that can be composed in the system, and in turn, the results in slower outputs than the Motorlab. However, both machines do demonstrate the effect derivative has in control systems.

Table 10.2: List of gains to be used in Motorlab's PD control experiment

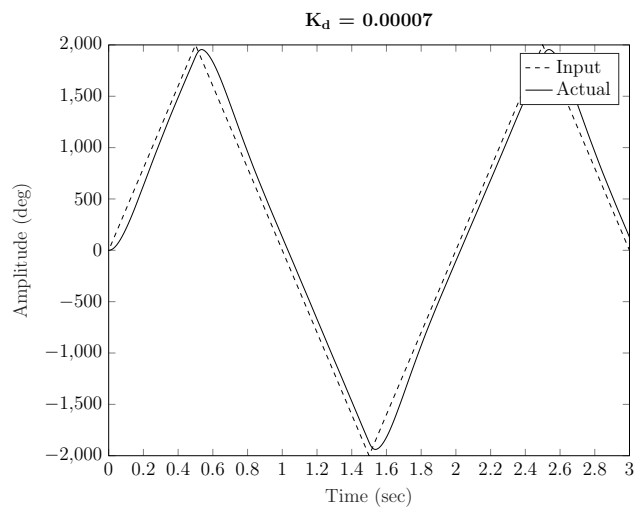
| K_p | z (rad/s) | Magnitude of Ramp (deg) |
|---------|-------------|----------------------------|
| 0.001 | none | 2000 |
| 0.00007 | 10 | 2000 |
| 0.001 | 10 | 2000 |



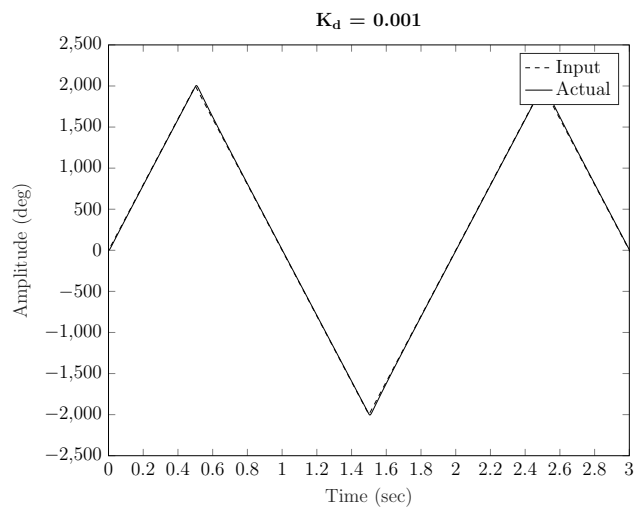
(a)



(b)



(c)



(d)

Figure 10.7: Motorlab PD Controller Experimental Results.

Chapter 11

Conclusion

Several experiments were explored on the NERMLAB to demonstrate its feasibility as laboratory hardware in control theory application. The feasibility was assessed by conducting similar procedures and experiments that are carried out in Control of Mechanical Systems at Kansas State University. The results of the experiments were compared to the results of laboratory hardware known as "Motorlab". This thesis addressed the concerns and limitations that arise when using more economical hardware for laboratory outcomes.

Chapter 5 demonstrated how a linear estimate of friction could be approximated on a motor. Additionally, it allows students to come up with model parameters for their own hardware and show how it compares to a theoretical model, such as a simple speed decay, which was demonstrated in this chapter. Chapter 6 modeled and experimented with a position control system. It effectively showed how simply changing the proportional gain of a system will not lead to system improvement, such as decreasing the settling time. Lastly, Chapters 7 and 9 attempted to address the concepts of high frequency dynamics and steady state error, respectively. These experiments suffered from encoder noise; however, in both cases, successfully demonstrated the overall concept that was introduced in each chapter. The developed models in these chapters were slightly more complicated than the models for the Motorlab system, but still simple enough that it would not overly complicate a student's

ability to understand and conduct experiments on the NERMLAB.

Inexpensive hardware did have drawbacks when comparing it to more expensive alternatives. For example, lower operating limits restricted the gain range a system could experience. Additionally, having noisy encoder output when doing speed control experiments obfuscate the learning objectives of the laboratory. However, these things certainly do lessen the impact on learning objectives achieved when comparing to the Motorlab; but, more economical hardware just means different objectives need to be considered when conducting experimentation. It provides institutions and instructors with new ideas and concepts to instruct upon, in addition to the theories, that rarely make it into textbooks.

Bibliography

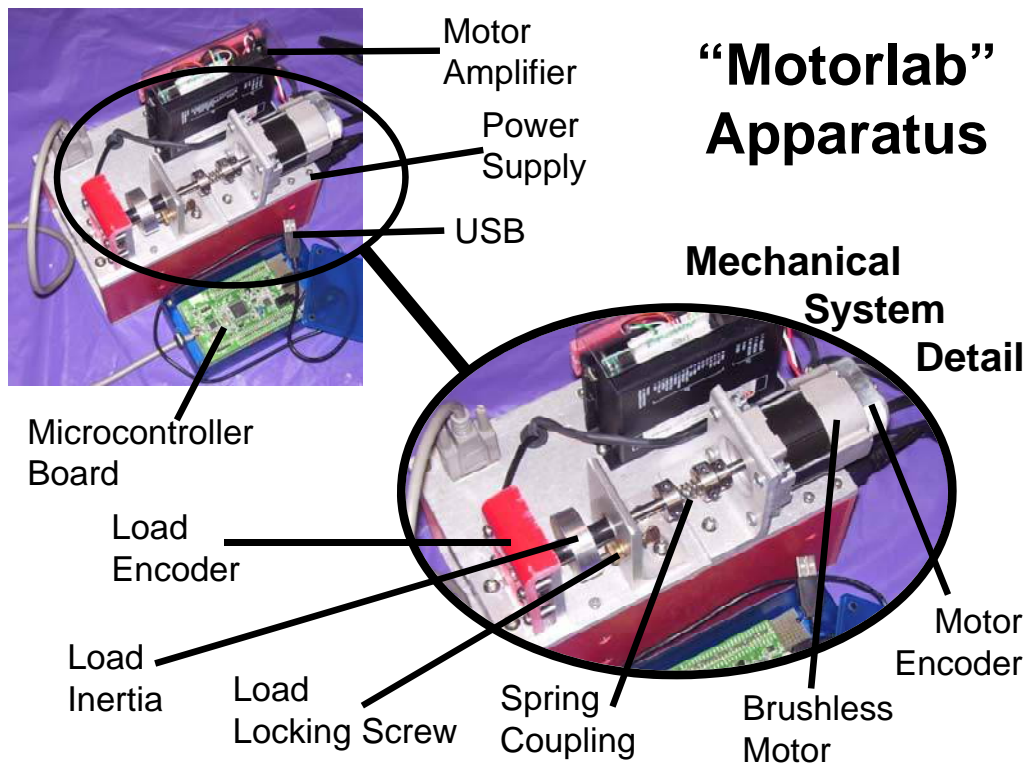
- [1] R. M. Reck and R. S. Screenivas, “Developing a new affordable dc motor laboratory kit for an existing undergraduate controls course,” in *American Control Conference (ACC)*, (Chicago, IL), pp. 2801–2806, 2015.
- [2] R. M. Reck, “Experiential learning in control systems laboratories and engineering project management,” dissertation, University of Illinois at Urbana-Champaign, 2016.
- [3] S. R. Smith, “Demonstrating introductory control systems concepts on inexpensive hardware,” Master’s thesis, Kansas State University, 2017.
- [4] S. Hendrix, “Take home labs and the furuta pendulum,” thesis, Oklahoma State University, 2012.
- [5] AMS, *AS5047D 14-Bit On-Axis Magnetic Rotary Position Sensor with 11-Bit Decimal and Binary Incremental Pulse Count*. AMS, April 2016.
- [6] J. R. Mevey, “Sensorless field oriented control of brushless permanent magnet synchronous motors,” Master’s thesis, Kansas State University, 2009.
- [7] D. Y. Ohm, *Dynamic Model of PM Synchronous Motors*. Drive Tech, 2000.
- [8] K. LIU and Z. ZHU, “Online estimation of the rotor flux linkage and voltage-source inverter nonlinearity in permanent magnet synchronous machine drives,” *IEEE Transactions on Power Electronics*, pp. 418–427, 2014.
- [9] R. M. Reck, “Defining common aspects of undergraduate instructional laboratories for control systems,” in *American Control Conference (ACC)*, (Boston, MA), pp. 6646–6651, 2016.

- [10] O. J. Oguntoyinbo, “Pid control of brushless dc motor and robot trajectory planning and simulation with matlab/simulink,” report, Vaasan Ammattikorkeakoulu Vasa Yrkeshogskola, University of Applied Sciences, 2009.
- [11] J.-W. Jung, “Space vector pwm inverter,” tech. rep., The Ohio State University, 2005.
- [12] R. M. Reck, R. S. Screenivas, and M. C. Loui, “Assessing an affordable and portable laboratory kit in an undergraduate control systems course,” in *Frontiers in Education Conference (FIE)*, (El Paso, TX), pp. 1–4, 2015.

Appendix A

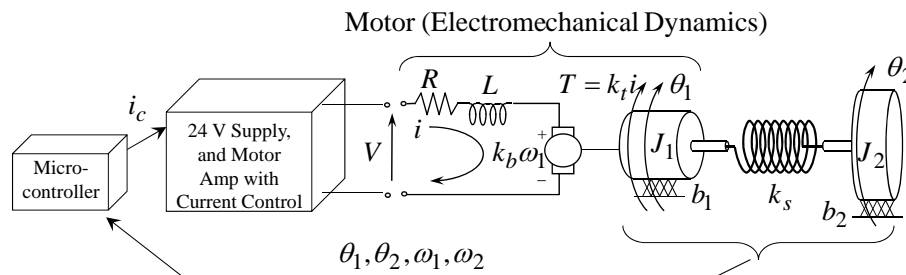
Motorlab Specifications

“Motorlab” Dynamics and Controls System

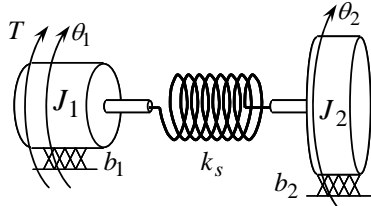


System Description

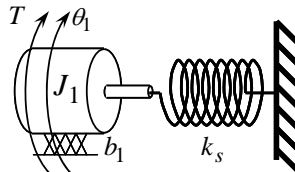
Below is a schematic representation of the Motorlab system in a closed-loop position or speed control configuration. There are two position sensors on the apparatus, a motor encoder and a load encoder. The speeds of the two inertias are measured by numerically differentiating the position signals in the computer controlling the system (microcontroller). The motor amplifier has a control loop that measures and controls the electric current in the motor windings. This results in what is commonly known as a “torque controlled” motor, since the magnetic torque is proportional to the current in the windings. The microcontroller is interfaced to the motor amplifier through a $\pm 10\text{V}$ analog signal. By varying the magnitude of this voltage the microcontroller can change the current in the motor. This voltage, which is proportional to the controlled current, serves as a current command (desired current) for the current control loop in the amplifier. An additional sensor, not shown below, is the current sensor in the amplifier used to implement the current control. The signal from this sensor is also read by the microcontroller, using an analog to digital converter. Although this signal is not used in the control loops on the microcontroller, it is recorded for data analysis.



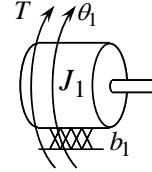
Several different configurations of the system can be utilized in experiments. Either sensor, the motor or load encoder, can be used for the feedback of the control loop. The selection is made in the software interface. The motor encoder is known as a “collocated” sensor since it is co-located with the input to the mechanical system, the motor torque. The load sensor is separated from the input to the system by a spring and is therefore known as a “non-collocated” sensor. In addition to varying which sensor is used, the mechanical system can be changed with the lock down screw and the spring coupling. Also, a choice can be made between velocity control or position control by selecting the appropriate control program. Any of the following mechanical models may be realized using the Motorlab hardware and software.



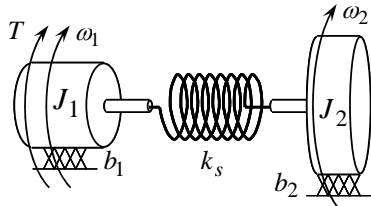
Fourth order system with a free integrator



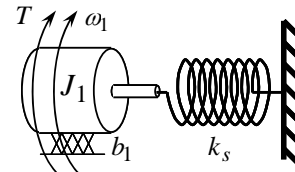
Second order system



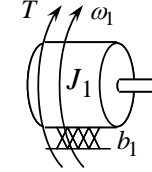
Second order system with a free integrator



Third order system



Second order system with a free differentiator



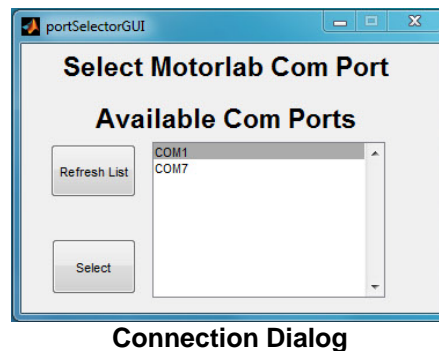
First order system

Software

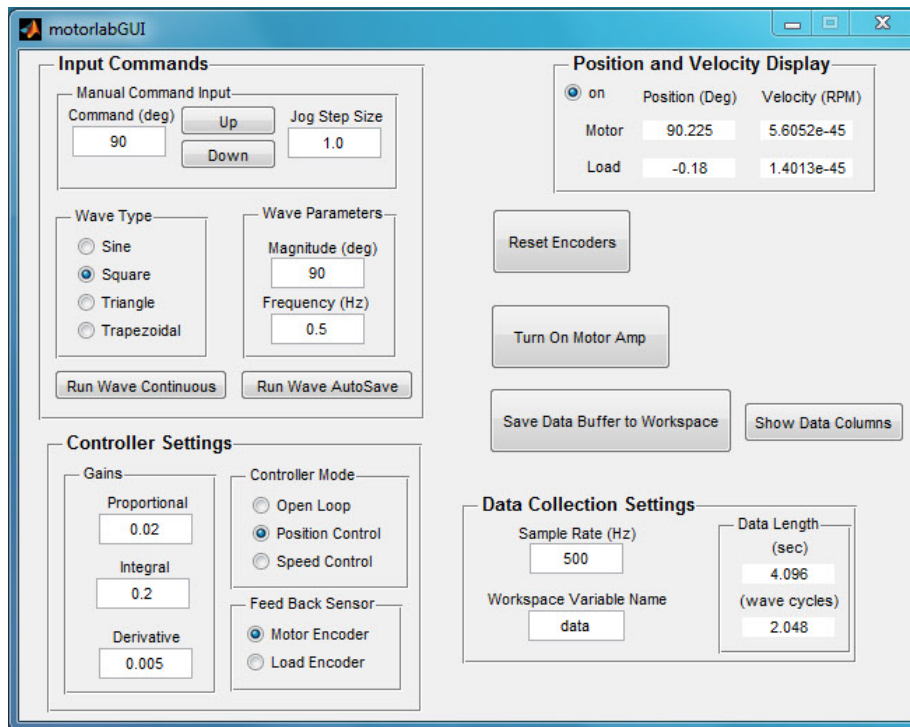
The software for the system can be found in the “c:\Motorlab” directory on the laboratory machines. All the needed Matlab functions can be found there. The software that is on the microcontroller is included in this directory in the motorlabRepo.zip file. This program is burned into the flash memory of the microcontroller and runs on power up. The software that runs on the PC is a GUI written in Matlab (“motorlabGUI.m”). There are additional m-files in the “Motorlab” directory that can be used to plot data from the system.

User Interface

To run the Motorlab GUI you must open Matlab and add the “c:\Motorlab” directory to the Matlab path or set this directory as the current directory. Normally you will add it to the path and set the current directory to the location where you are storing your files. The microcontroller should be plugged into USB. In the Matlab command window type “motorlabGUI.” The opening dialog (below) asks you to select the communication port for the microcontroller. If more than one port is listed you should be able to detect which is the Motorlab by unplugging the USB or powering it down and then clicking the “Refresh List” button. The GUI should open after selecting the com port.



Connection Dialog



Motorlab GUI

Data Acquisition

The microcontroller stores data in a circular buffer that is 2048 data samples in length with 9 variables in each sample. After 2048 sample periods the buffer begins to be overwritten with the more recent data. At any time the buffer contains the most recent 2048 samples. Pressing the "Save Data Buffer to Workspace" button will write this data to a 2048x9 matrix in the Matlab workspace. Pressing the "Run Wave AutoSave" button starts the wave type selected and then writes the data to the Matlab workspace once the buffer has filled with new data. The time length of the data depends on the sample rate. If for example the sample rate is set to 500 Hz, then the last 4.096 seconds (2048/500) of data will be saved in the buffer.

The data matrix saved in the Matlab workspace contains 9 variables (columns). The ninth column is reserved. The other eight are listed below. Note that the variable in the second column changes. It depends on the "Controller Mode" chosen at the time of the data storage.

| Column | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-------------|-----------|-------------------------------------------------------|------------------|------------------|------------------|------------------|-----------------|---------------|
| Description | Time | Command | Motor Encoder | Load Encoder | Motor Speed | Load Speed | Current Command | Motor Current |
| Variable | t (sec) | θ_c (deg), ω_c (rpm), i_c (Amp) | θ_1 (deg) | θ_2 (deg) | ω_1 (rpm) | ω_2 (rpm) | i_c (Amp) | i (Amp) |

M-files for plotting

There are m-files provided in the "c:\Motorlab" directory that can be used to plot the data from the Motorlab. Although you will frequently want the access the data with your own m-files, these files are useful for quickly viewing the data after acquiring it. There is one file for each of the "Controller Mode" settings.

File: *mlolplots.m* function: `mlolplots(data,Iscale);` Uses data generated by the Motorlab in open loop control. If an "Iscale" argument is supplied then the commanded current values are scaled by the Iscale value in the plots.

example: `mlolplots(data);` Does not scale the current command.

example: `mlolplots(data,Iscale);` Multiplies commanded current values by Iscale.

File: mlposplots.m function: mlposplots(data); Uses data generated by the Motorlab position control mode.
example: mlposplots(data);

File: mlspeedplots.m function: mlspeedplots(data); Uses data generated by the Motorlab velocity control mode.
example: mlvelplots(data);

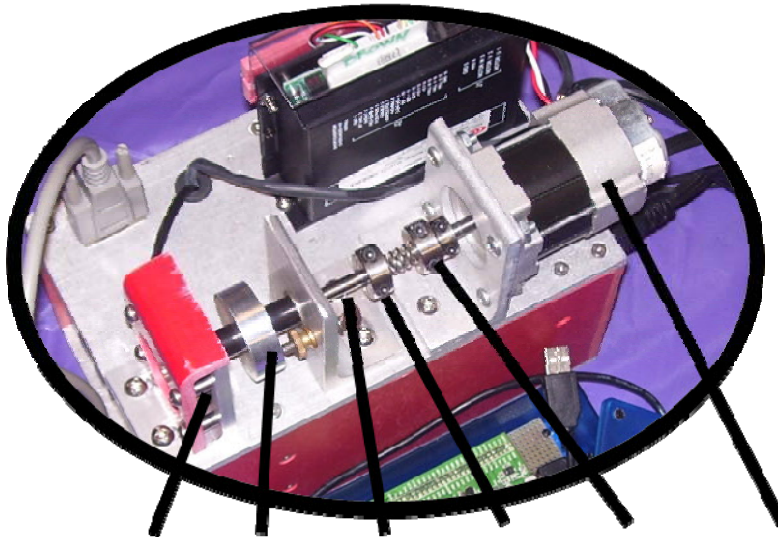
File: trapprof.m function: [x,v,t]=trapprof(DX,Vmax,Amax,DT) Trapezoidal-velocity motion profile generation
Outputs: x=position vector, v=trapezoidal velocity vector, t=time vector
Inputs: DX=distance to move, Vmax=maximum velocity, Amax=maximum acceleration, DT=time step for outputs
example: [x,v,t]=trapprof(DX,Vmax,Amax,DT)

Hardware Specifications

Important Scaling Considerations

- Motor Amplifier Scaling = 1 Amp/Volt. Therefore, one Volt output from the microcontroller corresponds to a one Amp command to the current control loop in the motor amplifier. The plotting routines provided take this scaling into consideration.
- Position is measured in degrees and velocity is measured in RPM. The output of the control algorithm in the microcontroller is measured in Volts. Therefore, for example, the units of the proportional and derivate gains in the position controller would be Volts/deg and Volts*sec/deg, respectively. When multiplied by the amplifier scaling (1 Amp/Volt) these gains become Amp/deg and Amp*sec/deg. The units of the proportional gain in the velocity controller would be Volts/RPM (or Amp/RPM if amplifier scaling is included).

Inertias



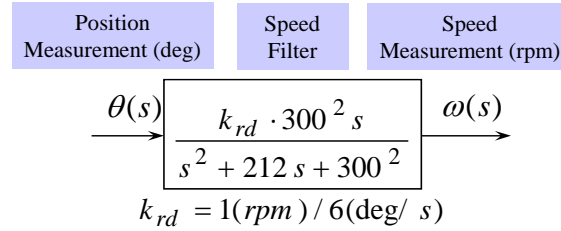
| Object | Load Encoder | Load Wheel | Load Shaft | Single Shaft Collar | Double Shaft Collar | Motor Rotor and Motor Encoder |
|-----------------------------|--------------|------------|------------|---------------------|---------------------|-------------------------------|
| Inertia(g-cm ²) | 0.8 | 82 | 1.4 | 15 | 19 | 110 |

A Few Other Details

- Max Data Acquisition Sample Rate = 10 kHz (the control update rate of the microcontroller software)
- Motor Encoder Resolution = 360 deg/1600 counts = 0.225 deg/count
- Load Encoder Resolution = 360 deg/2000 counts = 0.18 deg/count
- Max motor velocity with the 24 Volt power supply is about 4000 rpm

Speed Measurement

The two speeds measured by the Motorlab system are found using a discrete time approximation (i.e. computer code) of a derivative with a low pass filter. The continuous time transfer function for this filter is given below. It uses the encoder position measurement for input. Note the free s in the numerator performs the differentiation and the filter with a cutoff frequency of 300 rad/s helps to filter spikes in the speed measurement caused by differentiating the discrete steps inherent in an encoder position measurement.



Specs from Motor Manufacturer's Data Sheet

| LA052-040E Motor Dynamic Specs From Shinano Kenshi | | |
|----------------------------------------------------|-------------------|-------|
| | UNITS | Value |
| RATED POWER | W | 40 |
| RATED VOLTAGE | VDC | 24 |
| RATED SPEED | rpm | 3,000 |
| RATED TORQUE | N-cm | 12.7 |
| | kgf-cm | 1.3 |
| RATED CURRENT | A | 2.5 |
| TORQUE CONSTANT | N-cm/A | 5.0 |
| | kgf-cm/A | 0.51 |
| BACK EMF CONSTANT | V/krpm | 5.2 |
| PHASE RESISTANCE | Ohm | 1.18 |
| PHASE INDUCTANCE | mH | 4.4 |
| INSTANTANEOUS PEAK TORQUE | N-cm | 38.2 |
| MAX SPEED | rpm | 5,000 |
| ROTOR INERTIA | g-cm ² | 110 |
| POWER RATE | kW/s | 1.48 |
| MECHANICAL TIME CONSTANT | ms | 5.2 |
| ELECTRICAL TIME CONSTANT | ms | 3.7 |
| MASS | kg | 0.6 |

Current Control Loop Model

The motor amplifier has a current control loop. As configured in the Motorlab apparatus this loop has a bandwidth of approximately 400 Hz. Using data acquired from step and sinusoidal responses the following two closed loop transfer functions have been identified as approximate models for the closed-loop current control dynamics.

$$T_i = \frac{\omega_n^2 (s+z)}{z(s^2 + 2\zeta\omega_n s + \omega_n^2)} \quad \text{or} \quad \begin{cases} T_{idelay} = \frac{\omega_n^2 (s+z)}{z(s^2 + 2\zeta\omega_n s + \omega_n^2)} e^{-t_d s} \\ T_{ipade} = \frac{\omega_n^2 (s+z)}{z(s^2 + 2\zeta\omega_n s + \omega_n^2)} \cdot \frac{s^2 - 6s/t_d + 12/t_d^2}{s^2 + 6s/t_d + 12/t_d^2} \end{cases}$$

where :

$$z = 170 \cdot 2\pi \text{ (rad/sec)}$$

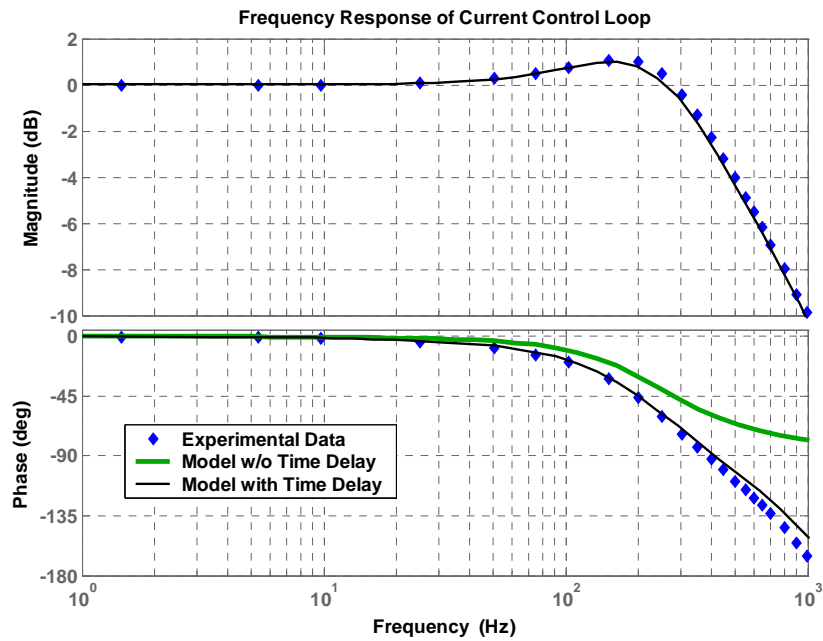
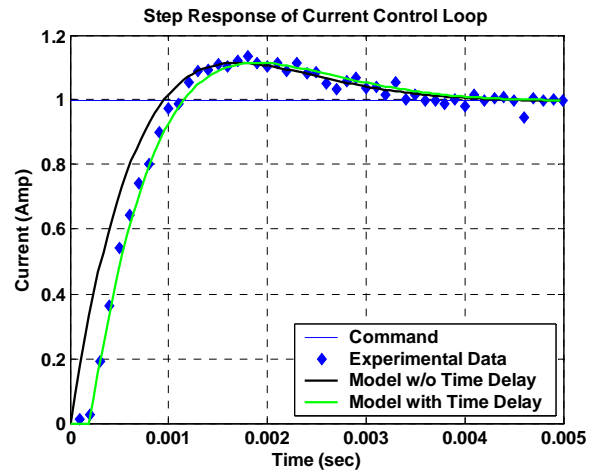
$$\omega_n = 230 \cdot 2\pi \text{ (rad/sec)}$$

$$\zeta = 0.8$$

$$t_d = 0.0002 \text{ (sec)}$$

Two of the models above contain a time delay while the other does not. One model with the time delay uses the exponential (exact) representation with the delay, while the other uses a second order Pade' approximation of the

delay. In the following two figures the responses of these two models are compared with actual data acquired from one of the Motorlab systems. Both the step response and the frequency response models are shown.



Motor Amplifier Manual

FEATURES

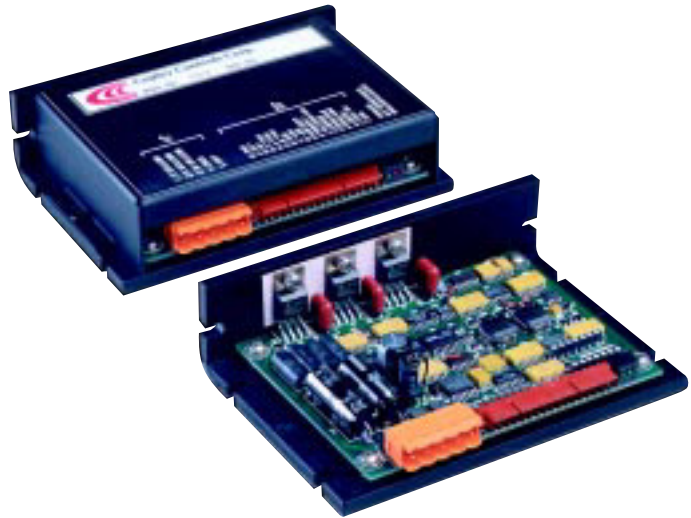
- *CE Compliance to 89/336/EEC*
- *Recognized Component to UL 508C*
- Complete torque (current) mode functional block
- Drives motor with 60° or 120° Halls
- Single supply voltage 18-55VDC
- 5A continuous, 10A peak more than double the power output of servo chip sets
- Fault protected
Short-circuits from output to output, output to ground
Over/under voltage
Over temperature
Self-reset or latch-off
- 2.5kHz bandwidth
- Wide load inductance range 0.2 to 40 mH.
- +5, +15V Hall power
- Separate continuous, peak, and peak-time current limits
- Surface mount technology

APPLICATIONS

- X-Y stages
- Robotics
- Automated assembly machinery
- Component insertion machines

THE OEM ADVANTAGE

- **NO POTS:** Internal component header configures amplifier for applications
- Conservative design for high MTBF
- Low cost solution for small brushless motors to 1/3 HP



PRODUCT DESCRIPTION

Model 503 is a complete pwm servoamplifier for applications using DC brushless motors in torque (current) mode. It provides six-step commutation of three-phase DC brushless motors using 60° or 120° Hall sensors on the motor, and provides a full complement of features for motor control. These include remote inhibit/enable, directional enable inputs for connection to limit switches, and protection for both motor and amplifier.

The /Enable input has selectable active level (+5V or gnd) to interface with most control cards.

/Pos and /Neg enable inputs use fail-safe (ground to enable) logic.

Power delivery is four-quadrant for bi-directional acceleration and deceleration of motors.

Model 503 features 500W peak power output in a compact package using surface mount technology.

An internal header socket holds components which configure the various gain and current limit settings to customize the 503 for different loads and applications.

Separate peak and continuous current limits allow high acceleration without sacrificing protection against continuous overloads. Peak current time limit is settable to match amplifier to motor thermal limits.

Header components permit compensation over a wide range of load inductances to maximize bandwidth with different motors.

Package design places all connectors along one edge for easy connection and adjustment while minimizing footprint inside enclosures.

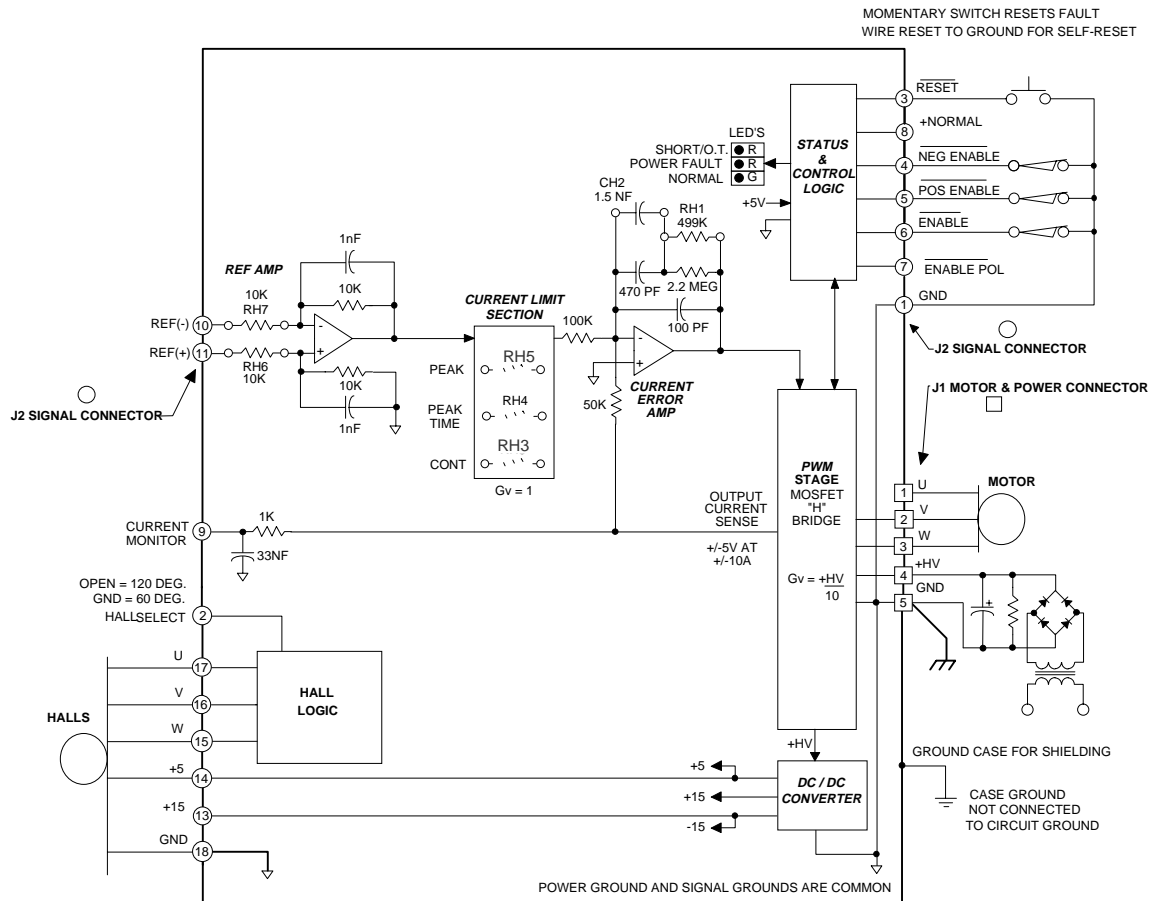
High quality components and conservative ratings insure long service life and high reliability in industrial installations.

A differential amplifier buffers the reference voltage input to reject common-mode noise resulting from potential differences between controller and amplifier grounds.

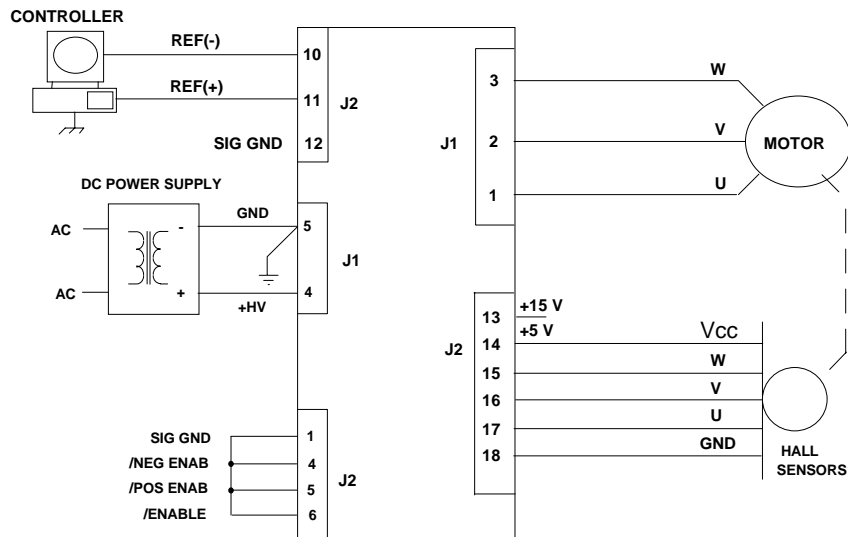
Output short circuits and heatplate overtemperature cause the amplifier to latch into shutdown. Grounding the reset input will enable an auto-reset from such conditions when this feature is desired.

Model 503 DC Brushless Servo Amplifier

FUNCTIONAL DIAGRAM



TYPICAL CONNECTIONS



APPLICATION INFORMATION

To use the model 503 set up the internal header with the components that configure the transconductance, current limits, and load inductance. Current-limits and load inductance set up the amplifier for your particular motor, and the transconductance defines the amplifiers overall response in amps/volt that is required by your system.

COMPONENT HEADER SETTINGS

Use the tables provided to select values for your load and system. We recommend that you use these values as starting points, adjusting them later based on tests of the amplifier in your application.

LOAD INDUCTANCE (RH1,CH2)

Maximizes the bandwidth with your motor and supply voltage. First replace CH2 with a jumper (short). Adjust the value of RH1 using a step of 1A or less so as not to experience large signal slew-rate limiting. Select RH1 for the best transient response (lowest risetime with minimal overshoot). Once RH1 has been set. choose the smallest value of CH2 that does not cause additional overshoot or degradation of the step response.

TRANSCONDUCTANCE (RH6,7)

The transconductance of the 503 is the ratio of output current to input voltage. It is equal to $10k\Omega/RH6$ (Amps/Volt). RH6, and RH7 should be the same value and should be 1% tolerance metal film type for good common-mode noise rejection.

CURRENT LIMITS (RH3, 4, & 5)

The amplifier operates at the 5A continuous, 10A peak limits as delivered. To reduce the limit settings, choose values from the tables as starting points, and test with your motor to determine final values. Limit action can be seen on current monitor when output current no longer changes in response to input signals. Separate control over peak, continuous, and peak time limits provides protection for motors, while permitting higher currents for acceleration.

SETUP BASICS

1. Set RH1 and CH2 for motor load inductance (see following section).
2. Set RH3, 4, & 5 if current limits below standard values is required.
3. Ground the /Enable (/Enable Pol open), /Pos Enable, and /Neg Enable inputs to signal ground.
4. Connect the motor Hall sensors to J2 based on the manufacturers suggested signal names. Note that different manufacturers may use A-B-C, R-S-T, or U-V-W to name their Halls. Use the required Hall supply voltage (+5 or +15V). *Note that there is a 30 mA limit at +5V. Encoders that put-out Hall signals typically consume 200-300 mA, so if these are used, then they must be powered from an external power supply.*
5. Connect J1-4,5 to a transformer-isolated source of DC power, +18-55V. Ground the amplifier and power supply with an additional wire from J1-5 to a central ground point.

6. With the motor windings disconnected, apply power and slowly rotate the motor shaft. Observe the Normal (green) led. If the lamp blinks while turning then the 60/120° setting is incorrect. If J2-2 is open, then ground it and repeat the test. In order to insure proper operation, the correct Hall phasing of 60° or 120° must be made.

6. Turn off the amplifier and connect the motor leads to J1-1,2,3 in U-V-W order. Power up the unit. Apply a sinusoidal reference signal of about 1 Hz. and 1Vrms between

Ref(+) and Ref(-), J2-10,11.

7. Observe the operation of the motor as the current monitor signal passes through zero. When phasing is correct the speed will be smooth at zero crossing and at low speeds. If it is not, then power-down and re-connect the motor.

There are six possible ways to connect the motor windings, and only one of these will result in proper motor operation. The six combinations are listed in the table below. Incorrect phasing will result in erratic operation, and the motor may not rotate. When the correct combination is found, record your settings.

| | J1-1 | J1-2 | J1-3 |
|----|------|------|------|
| #1 | U | V | W |
| #2 | V | W | U |
| #3 | W | U | V |
| #4 | U | W | V |
| #5 | W | V | U |
| #6 | V | U | W |

GROUNDING & POWER SUPPLIES

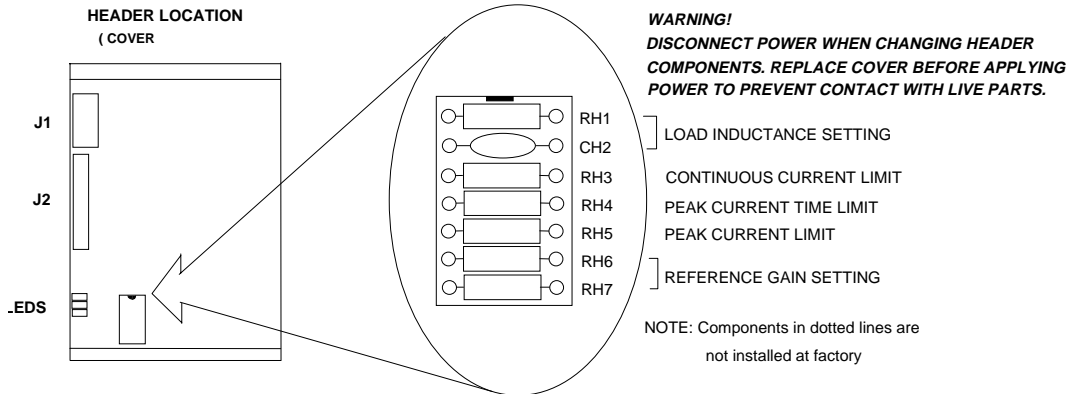
Power ground and signal ground are common (internally connected) in this amplifier. These grounds are isolated from the amplifier case which can then be grounded for best shielding while not affecting the power circuits.

Currents flowing in the power supply connections will create noise that can appear on the amplifier grounds.

This noise will be rejected by the differential amplifier at the reference input, but will appear at the digital inputs. While these are filtered, the lowest noise system will result when the power-supply capacitor is left floating, and each amplifier is grounded at its power ground terminal (J1-5). In multiple amplifier configurations, always use separate cables to each amplifier, twisting these together for lowest noise emission. Twisting motor leads will also reduce radiated noise from pwm outputs. If amplifiers are more than 1m. from power supply capacitor, use a small (500-1000μF.) capacitor across power inputs for local bypassing.

APPLICATION INFORMATION (CONT'D)

COMPONENT HEADER



CONTINUOUS CURRENT LIMIT (RH3)

| I _{cont} (A) | RH3 (Ω) |
|-----------------------|---------------|
| 5 | open * |
| 4 | 20k |
| 3 | 8.2k |
| 2 | 3.9k |
| 1 | 1.5k |

INPUT TO OUTPUT GAIN SETTING (RH6, RH7)

Note 1

Example: Standard value of RH6 is 10kΩ, thus G = 1 A/V

PEAK CURRENT LIMIT (RH5) Note 3

| I _{peak} (A) | RH5 (Ω) |
|-----------------------|---------------|
| 10 | open * |
| 8 | 12k |
| 6 | 4.7k |
| 4 | 2k |
| 2 | 750 |

LOAD INDUCTANCE SETTING (RH1 & CH2) Note 2

| Load (mH) | RH1 | CH2 |
|-----------|--------------|-----------------|
| 0.2 | 49.9 k | 1.5 nF |
| 1 | 150 k | 1.5 nF |
| 3 | 499 k | 1.5 nF * |
| 10 | 499 k | 3.3 nF |
| 33 | 499 k | 6.8 nF |
| 40 | 499 k | 10 nF |

PEAK CURRENT TIME-LIMIT (RH4) Note 4

| T _{peak} (s) | RH4 (Ω) |
|-----------------------|---------------|
| 0.5 | open * |
| 0.4 | 10 M |
| 0.2 | 3.3 M |
| 0.1 | 1 M |

Times shown are for 10A step from 0A

Notes: * **Standard values installed at factory are shown in italics.**

1. RH6 & RH7 should be 1% resistors of same value.
2. Bandwidth and values of RH1, CH2 are affected by supply voltage and load inductance. Final selection should be based on customer tests using actual motor at nominal supply voltage.
3. Peak current setting should always be greater than continuous current setting.
4. Peak times will double when current changes polarity. Peak times decrease as continuous current increases.

TECHNICAL SPECIFICATIONS

Typical specifications @ 25°C ambient, +HV = +55VDC. Load = 200μH. in series with 1 ohm unless otherwise specified.

OUTPUT POWER

| | |
|------------------------|---------------------------------|
| Peak power | |
| Unidirectional | ±10A @ 50V for 0.5 second, 500W |
| After direction change | ±10A @ 50V for 1 second, 500W |
| Continuous power | ±5A @ 50V, 250W |

OUTPUT VOLTAGE

$V_{out} = 0.97HV - (0.4)(I_{out})$

MAXIMUM CONTINUOUS OUTPUT CURRENT

| | |
|---------------------------------------------------------------|--------------------|
| Convection cooled, no conductive cooling | ±2A @ 35°C ambient |
| Mounted on narrow edge, on steel plate, fan-cooled 400 ft/min | ±5A @ 55°C |

LOAD INDUCTANCE

| | |
|---------------------------------------------|------------------------------------------------------------------|
| Selectable with components on header socket | 200 μH to 40mH (Nominal, for higher inductances consult factory) |
|---------------------------------------------|------------------------------------------------------------------|

BANDWIDTH

| | |
|-------------------------------------------------------------------------------------------------------|-------------------------------|
| Small signal | -3dB @ 2.5kHz with 200μH load |
| Note: actual bandwidth will depend on supply voltage, load inductance, and header component selection | |

PWM SWITCHING FREQUENCY

25kHz

ANALOG INPUT CHARACTERISTICS

| | |
|-----------|--------------------------------------------------------------|
| Reference | Differential, 20K between inputs with standard header values |
|-----------|--------------------------------------------------------------|

GAINS

| | |
|------------------------------|------------------------------------------------------------|
| Input differential amplifier | X1 as delivered. Adjustable via header components RH6, RH7 |
| PWM transconductance stage | 1 A/V (output vs. input to current limit stage) |

OFFSET

| | |
|-----------------------------------------|--------------------------------------------------|
| Output offset current (0 V at inputs) | 20 mA max. (0.2% of full-scale) |
| Input offset voltage | 20 mV max (for 0 output current, RH6,7 = 10kΩ) |

LOGIC INPUTS

| | |
|-------------------------------|----------------------------------------------------------------------------|
| Logic threshold voltage | HI: ≥ 2.5V , LO: ≤ 1.0V, +5V Max on all logic inputs |
| /Enable | LO enables amplifier (/Enable Pol open) , HI inhibits; 50 ms turn-on delay |
| /POS enable, /NEG enable | LO enables positive and negative output currents, HI inhibits |
| /Reset | LO resets latching fault condition, ground for self-reset every 50 ms. |
| /Enable Pol (Enable Polarity) | LO reverses logic of /Enable input only (HI enables unit, LO inhibits) |

LOGIC OUTPUTS

| | |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| +Normal | HI when unit operating normally, LO if overtemp, output short, disabled, or power supply (+HV) out of tolerance HI output voltage = 2.4V min at -3.2 mA max., LO output voltage = 0.5V max at 2 mA max. Note: Do not connect +Normal output to devices that operate > +5V |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

INDICATORS (LED's)

| | |
|----------------------|--------------------------------------------------------------------|
| Normal (green) | ON = Amplifier enabled, no shorts or overtemp, power within limits |
| Power fault (red) | ON = Power fault: +HV < 18V OR +HV > 55V |
| Short/Overtemp (red) | ON = Output short-circuit or over-temperature condition |

CURRENT MONITOR OUTPUT

±5V @ ±10A (2A/volt), 10kΩ, 3.3nF R-C filter

DC POWER OUTPUTS

| | |
|---------------------------------------------------|----------------------------------------|
| +5VDC | 30mA (Includes power for Hall sensors) |
| +15VDC | 10mA |
| Total power from all outputs not to exceed 200mW. | |

PROTECTION

| | |
|-----------------------------------------------------------|----------------------------------------------------------------|
| Output short circuit (output to output, output to ground) | Latches unit OFF (self-reset if /RESET input grounded) |
| Overtemperature | Shutdown at 70°C on heatplate (Latches unit OFF) |
| Power supply voltage too low (Undervoltage) | Shutdown at +HV < 18VDC (operation resumes when power > 18VDC) |
| Power supply voltage too high (Overvoltage) | Shutdown at +HV > 55VDC (operation resumes when power < 55VDC) |

POWER REQUIREMENTS

| | |
|-----------------------------------------------|------------------------|
| DC power (+HV) | +18-55 VDC @ 10A peak. |
| Minimum power consumption | 2.5 W |
| Power dissipation at 5A output, 55VDC supply | 10W |
| Power dissipation at 10A output, 55VDC supply | 40W |

THERMAL REQUIREMENTS

| | |
|-----------------------------|---------------------------------|
| Storage temperature range | -30 to +85°C |
| Operating temperature range | 0 to 70°C baseplate temperature |

MECHANICAL

| | |
|--------|------------------------------------------|
| Size | 3.27 x 4.75 x 1.28 in. (83 x 121 x 33mm) |
| Weight | 0.52 lb (0.24 kg.) |

CONNECTORS

| | |
|----------------|----------------------------------------------------|
| Power & motor | Weidmuller: BL-125946; Phoenix: MSTB 2.5/5-ST-5.08 |
| Signal & Halls | Molex: 22-01-3167 housing with 08-50-0114 pins |

***Excerpt of Data Sheet for the
STM32F4 Microcontroller***



UM1472 User Manual

STM32F4DISCOVERY STM32F4 high-performance discovery board

Introduction

The STM32F4DISCOVERY helps you to discover the STM32F4 high-performance features and to develop your applications. It is based on an STM32F407VGT6 and includes an ST-LINK/V2 embedded debug tool interface, ST MEMS digital accelerometer, ST MEMS digital microphone, audio DAC with integrated class D speaker driver, LEDs, pushbuttons and an USB OTG micro-AB connector.

Figure 1. STM32F4DISCOVERY

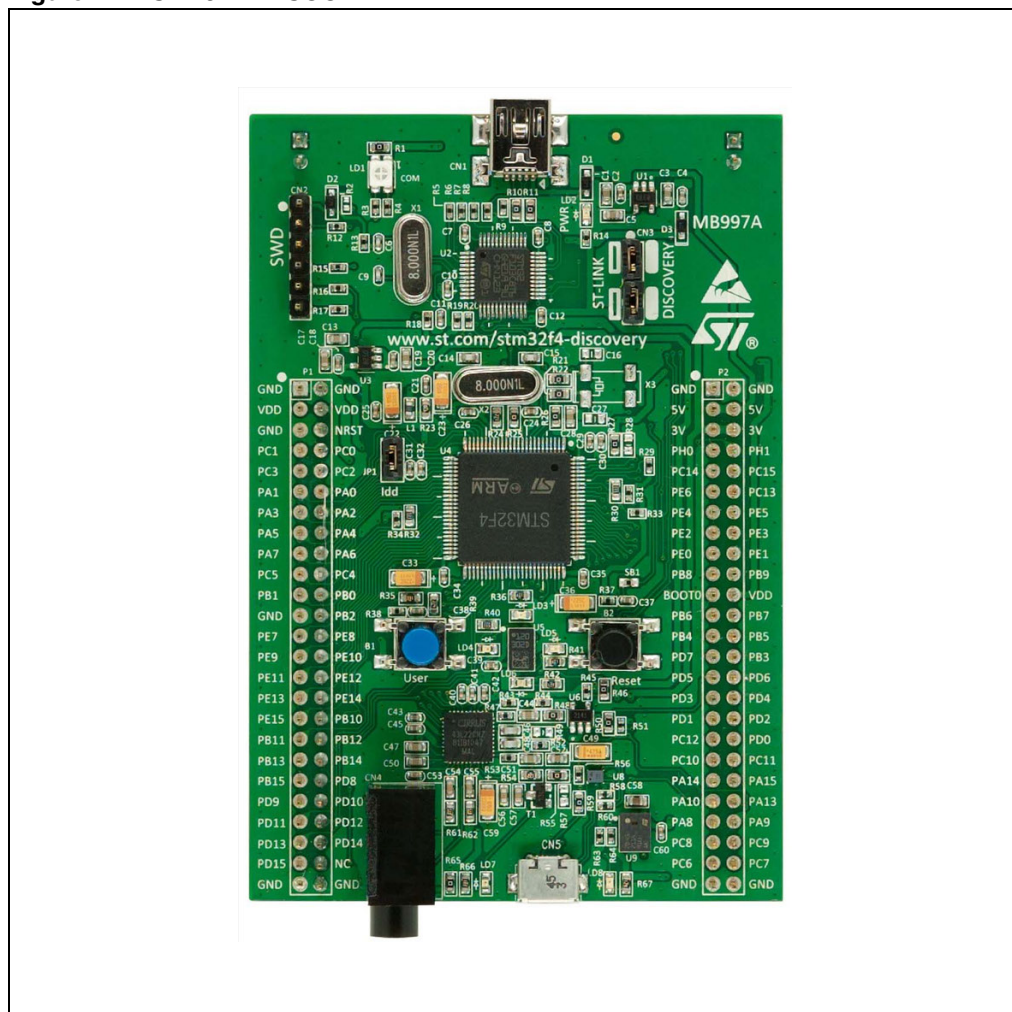
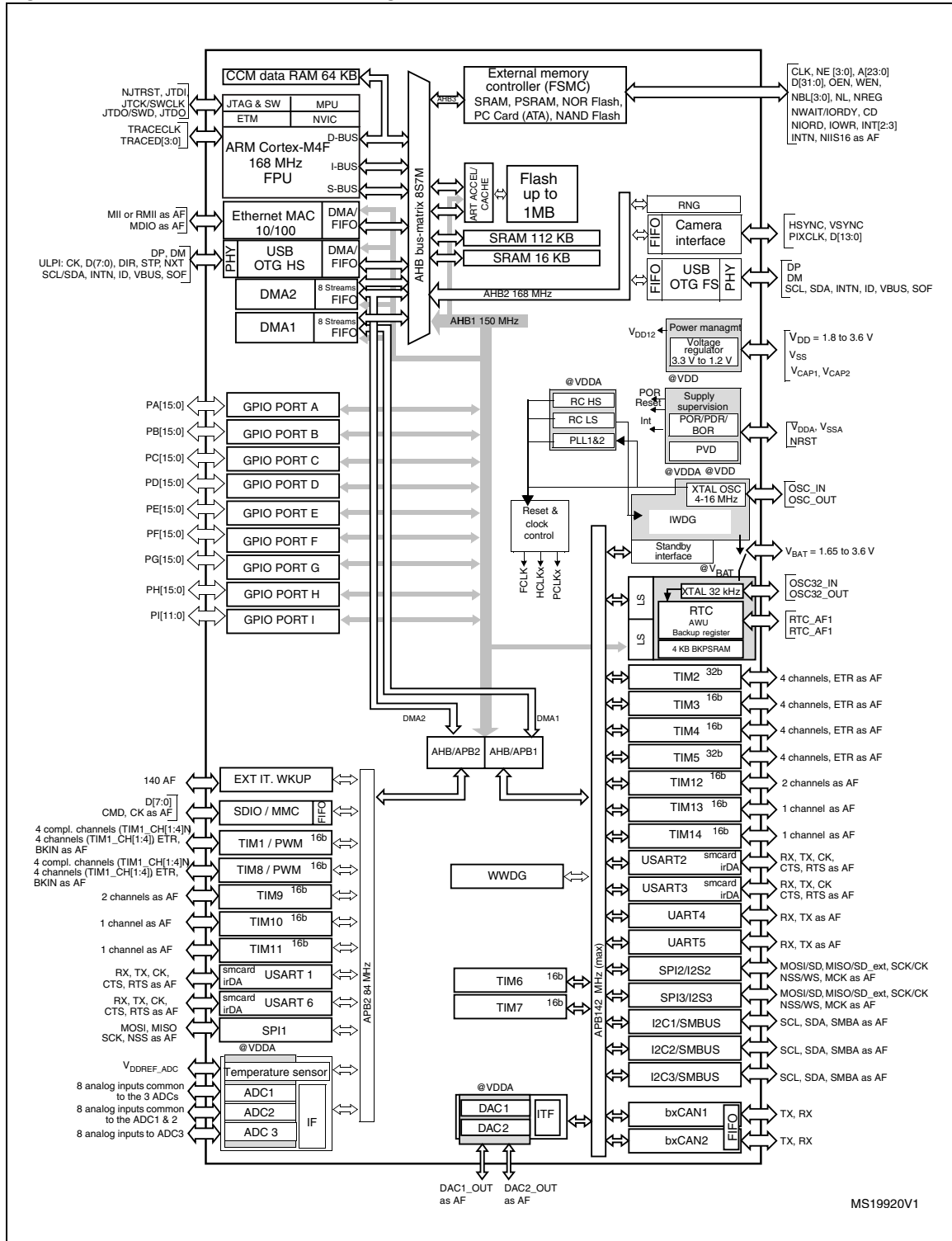


Figure 6. STM32F407VGT6 block diagram

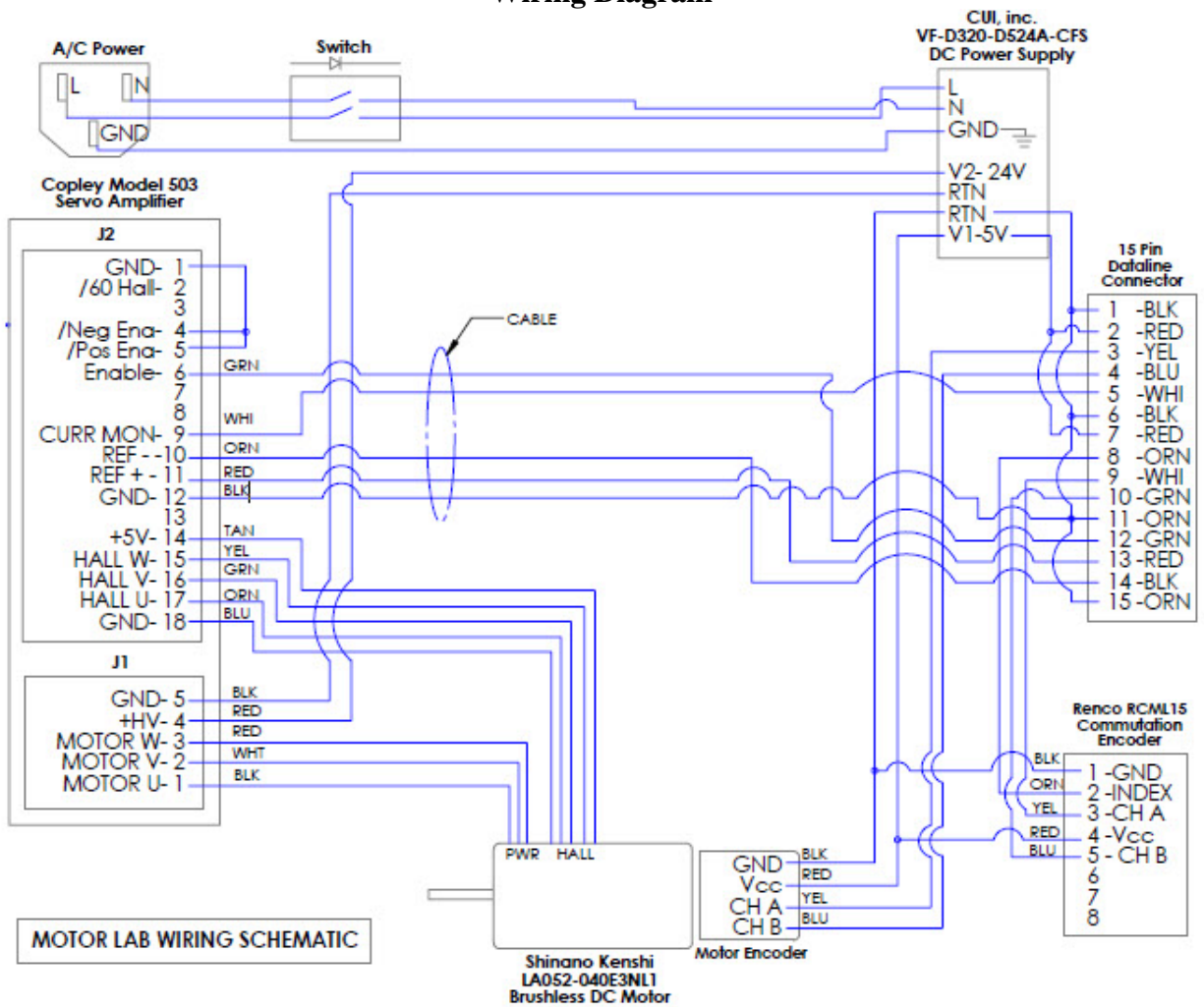


Wiring for the “Motorlab” Apparatus

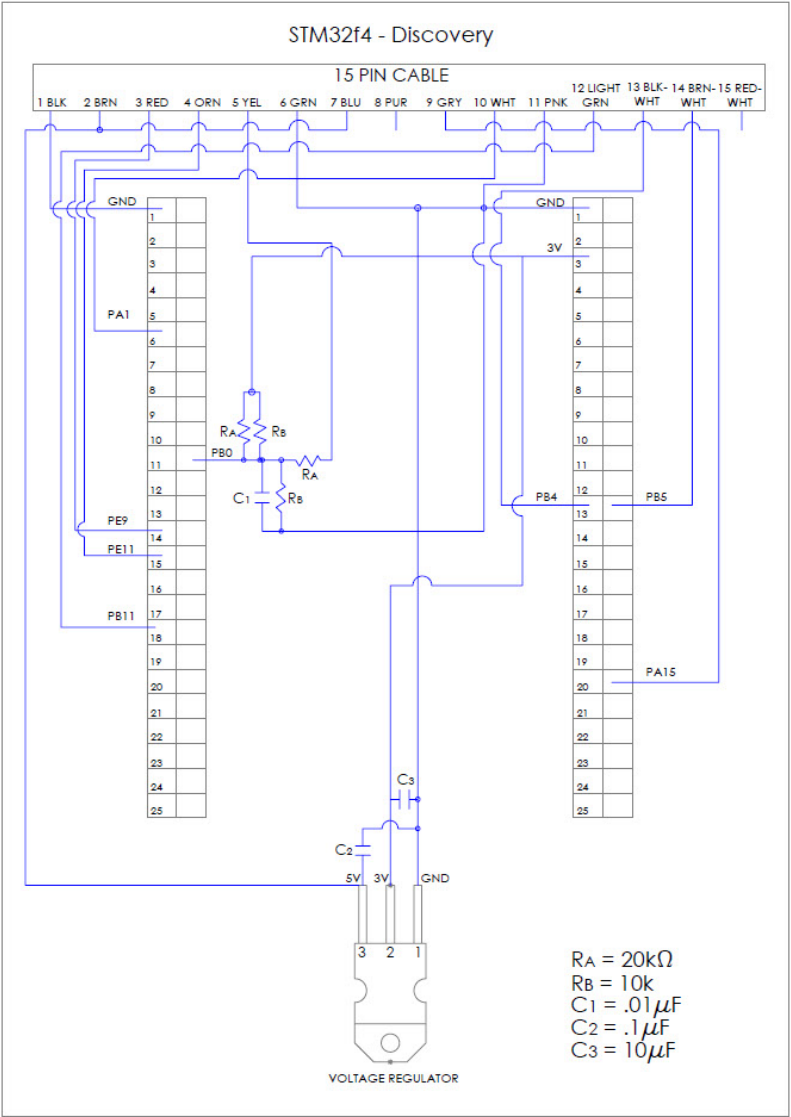
15 Pin Microcontroller Connections

| Amp J2 | Hardware Wire Color | 15pin Connector | Function | 15pin Cable | STM32f4 Discovery |
|----------|---------------------|-----------------|---------------------------|----------------|--------------------------------------|
| | Black | 1 Black | Motor Encoder Ground | 1 Black | GND |
| | Red | 2 Red | +5V Power | 2 Brown | Voltage Regulator |
| | Yellow | 3 Yellow | Motor Encoder Channel A | 3 Red | PE9 (TIM1-Ch1) |
| | Blue | 4 Blue | Motor Encoder Channel B | 4 Orange | PE11 (TIM1-Ch2) |
| Curr Mon | 9 White | 5 White | Current Monitor | 5 Yellow | PB0 (ADC1-Ch8) thru resistor network |
| | Black | 6 Black | Inertia Encoder Ground | 6 Green | GND |
| | Red | 7 Red | +5V Power | 7 Blue | Voltage Regulator |
| | Orange | 8 Orange | Inertia Encoder Index | 8 Purple | -- |
| | Yellow | 9 White | Inertia Encoder Channel A | 9 Gray | PA15 (TIM2-Ch1) |
| | Blue | 10 Green | Inertia Encoder Channel B | 10 White | PA1 (TIM2-Ch2) |
| GND | 12 Black | 11 Orange | Amp Signal Ground | 11 Pink | GND |
| Enable | 6 Green | 12 Green | Amp Signal (Amp Enable) | 12 Light Green | PB11 (Digital Out) |
| Ref+ | 11 Red | 13 Red | Current Command + | 13 Black-White | PB4 (TIM3-Ch1) |
| Ref- | 10 Orange | 14 Black | Current Command - | 14 Brown-White | PB5 (TIM3-Ch2) |
| GND | 12 Black | 15 Orange | GND | 15 Red-White | -- |

Wiring Diagram



STM32F4Discovery Host Board



Appendix B

Part Drawings

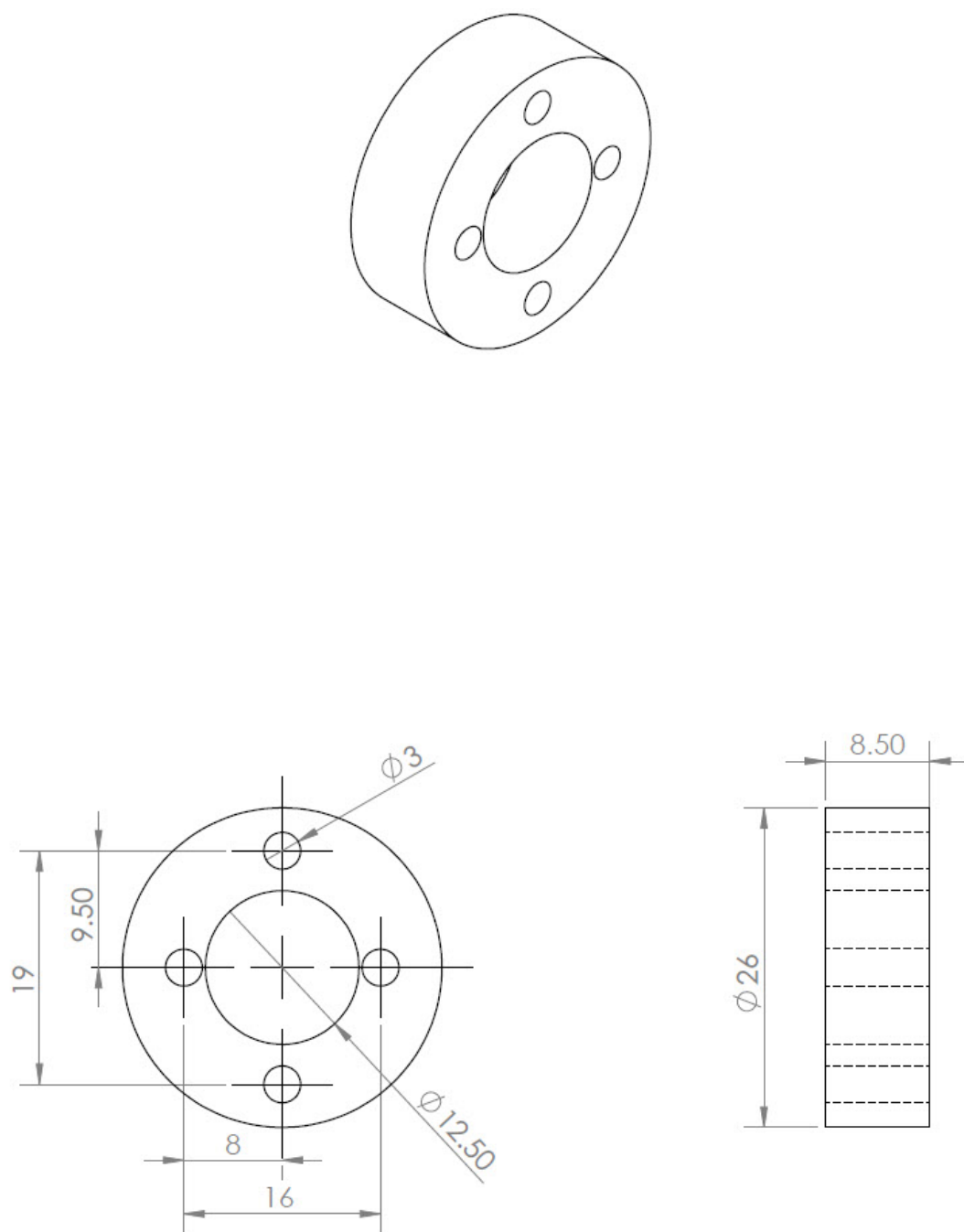


Figure B.1: Standoff for NERMLAB [mm]

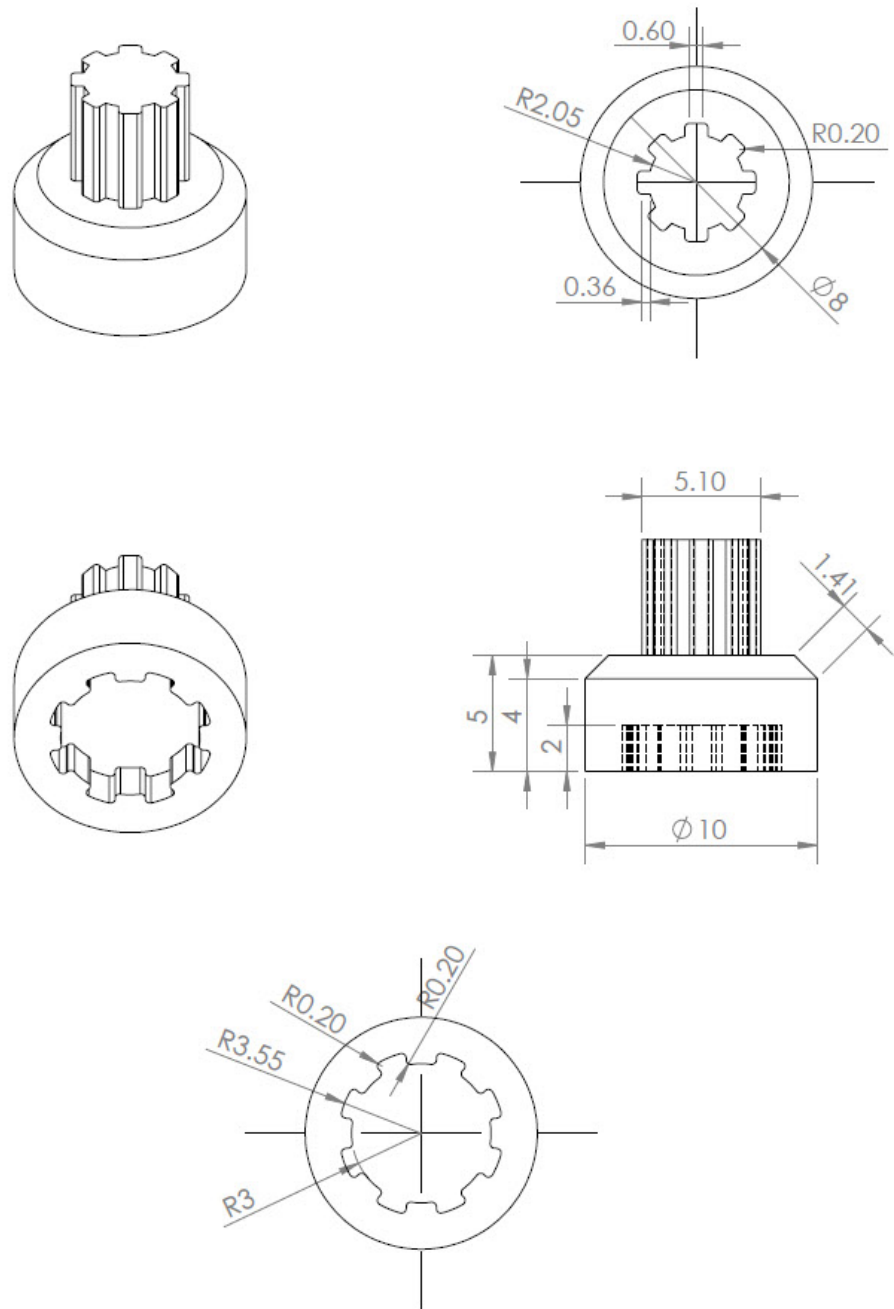


Figure B.2: Magnet holder for NERMLAB [mm]

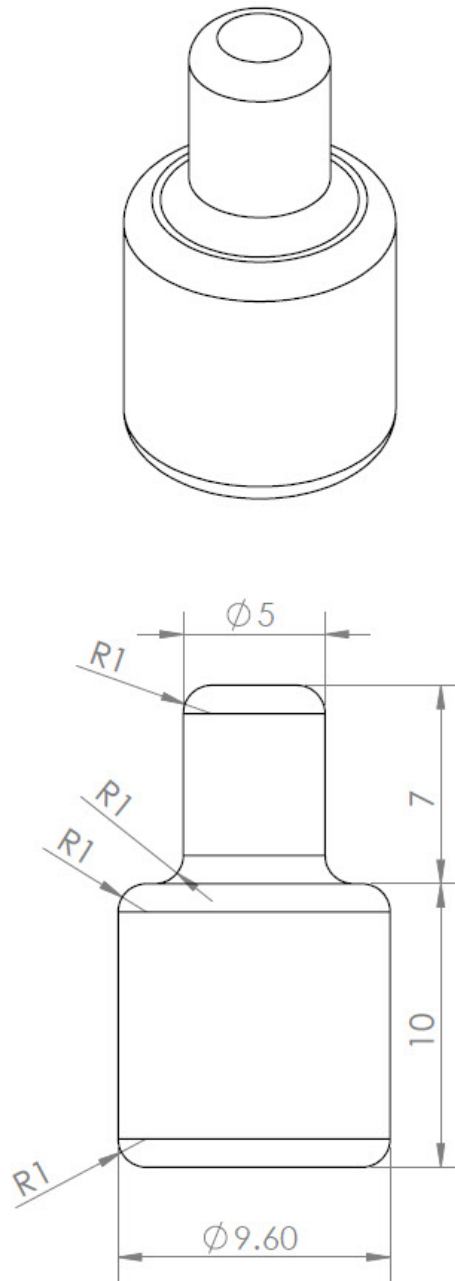


Figure B.3: Torque Transmission Shaft for NERMLAB [mm]

Appendix C

Laboratory Procedures

Appendix C includes the laboratory procedures that students carry out in Control of Mechanical Systems at Kansas State University.

Laboratory #2

In this lab you are to experimentally determine an approximation for the viscous friction coefficient for the motor of the Motorlab. Then you are to use this coefficient to predict the response of the dynamic system to an initial condition (initial velocity) and compare this to the actual I.C. response. If the spring coupling is removed from the Motorlab apparatus then we are left with the dynamic system described by the equations and schematic model to the right. This is the dynamic system studied in this laboratory. Also in this lab you will be continue to learn to use MATLAB.

ONE REPORT is due from each group, but you all are responsible for understanding what is in the report and how it was generated.

You are to set the “Controller Mode” in the motorlab GUI to “Open Loop” to acquire experimental data for this lab. This program does not implement closed-loop control of the Motorlab mechanical hardware. It allows you to manipulate the input to the mechanical system, the motor current (torque), and acquire data.

You are also to complete the MATLAB m-file that will generate the plots required for this lab. You are given most of the m-file code on the following page. You may copy this code out of this document and past it into an m-file to modify it. You should use the help in MATLAB and your instructor to continue to learn the language.

Estimating the Viscous Friction Coefficient

Looking at the differential equation in the model it can be seen that if a constant torque (current) is input then in steady state, where $\dot{\omega}(t) = 0$, $T(t) = b\omega(t)$. Therefore we should be able to estimate the viscous friction coefficient by obtaining steady state velocity and current data. Change the motor current command using the jog buttons then save the data to the workspace after sufficient time for the buffer to fill. Use the following two commands in the command window to get the average current and speed:

```
mean(data(:,8))
mean(data(:,5))
```

Fill in the table below. In this lab you will probably discover that friction is often a hard thing to model and that the linear, viscous friction, model is not completely accurate in some cases. We are attempting to find an "engineering estimate" that might be used in closed loop control where completely accurate models are not necessary.

| | | | | | | | | | | | | | |
|---------------------|-------|-------|-------|-------|-------|-------|---|------|------|------|------|------|------|
| Current Command (A) | -0.14 | -0.12 | -0.10 | -0.08 | -0.06 | -0.03 | 0 | 0.03 | 0.06 | 0.08 | 0.10 | 0.12 | 0.14 |
| Avg. Current (A) | | | | | | | | | | | | | |
| Avg. Speed (rpm) | | | | | | | | | | | | | |

Using the data from the table above obtain a plot of torque vs. angular velocity. On this same plot, draw the “best fit” line through the data by playing with the slope of the line described by $T = b_{estimate} \omega$. Generate this plot by completing the top of the m-file provided and running it with successive guesses at $b_{estimate}$.

Comparing Theoretical and Experimental IC Responses

Now that you have an estimate of b YOU ARE TO USE IT TO SOLVE THE DESCRIBING DIFFERENTIAL EQUATION given that $T(t) = 0$ and given that the initial condition (IC) is $\omega(t=0) = \omega_0$. Do this by hand, paying close attention to units. You are also to obtain data for the IC response from the actual system, and then compare this with the theoretical responses on the same plot. To obtain the response to an IC use a current of 0.15 A to generate an initial velocity. Using a sample frequency of 500 Hz do the following:

1. Jog the current to the desired level, or type it into the command window, and allow the velocity to settle, and wait at least four seconds for the data buffer to fill.
2. Hit the “Turn Off Motor Amp” button to turn off the current – count to 3 seconds – then immediately hit the “Save Data Buffer to Workspace” button. This should give data with about one second of initial velocity and 3 seconds of IC response.

$$T(t) = J\dot{\omega}(t) + b\omega(t)$$

$$T(t) = k_t i(t)$$

$$J = \text{inertia} = 129 \text{ g} \cdot \text{cm}^2$$

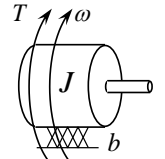
$$b = \text{friction coefficient} = \text{????}$$

$$k_t = \text{motor torque constant} = 5 \text{ N} \cdot \text{cm} / \text{A}$$

$$T(t) = \text{torque}$$

$$i(t) = \text{motor current}$$

$$\omega(t) = \text{angular speed}$$



3. In MATLAB and view the data with “mlolplots()”. In figure (2) of MATLAB you should see the IC response. Zoom in and use the cursor to find your actual initial velocity and a time at which the torque was set to zero.
4. Complete the bottom part of the m-file provided so that it generates the experimental response and the theoretical response on the same graph.

Things to Turn In

- You should have two different plots: 1) torque vs. angular velocity, 2) IC response
- You should have code for the m-file completed.
- Hand development of the solution to the differential equation with the I.C.
- Fill in the blanks below. **(This must be turned into your instructor before they leave the lab or other arrangements made with them. Attach another copy, which may be different/corrected, to your report).**

Fill In The Blanks: All answers should have units where appropriate. This is your chance to learn. So think about your answers, find as many connections as you can, and try to extrapolate. **You should copy this out of the given word file and fill in the blanks with BOLD face type and underlined.**

Lab #2 QUESTIONS Names

In the first plot we can see the relationship between motor speed and the friction torque. With a constant motor current, the motor **??????** is constant, and the speed settles to a fairly constant value where the input torque balances with the **??????** torque. Our typical model of friction for control design is **??????**, with the friction torque being proportional to velocity. However, the data points in the plot show that the actual friction has a **??????** component along with the linear component, resulting in a zero velocity with small values of constant torque. Our estimate of the viscous (linear) coefficient of friction roughly capturing both of these effects is **??????(units).**

In the second plot we see the actual initial condition response along with one from the model, which was found from the solution of the **??????** equation with an initial condition. The time constant the linear model can be found with the mass moment of inertia and our estimate of the friction coefficient. It has a numerical value of **??????(units).** At one time constant the linear is model is at exactly **??????(rpm)** (ignoring roundoff), which is 37% of the initial value. The actual data from system is at a value of **??????(rpm)** at the time constant. The major difference between the linear model and the actual system is at **??????** speed where the model significantly underestimates the friction torque. It can be seen that as the motor slows down the velocity decay is much faster than predicted by the **??????**.

Starting m-file code

```
% lab 2 starting file
i=[-0.14 -0.12 -0.10 -0.08 -0.06 -0.03 0 0.03 0.06 0.08 0.10 0.12 0.14]; % YOU SHOULD CHANGE THIS
% TO THE AVERAGE VALUES

rpm=[????] %vector of velocity data

kt = ???;
T=kt*i; % convert current data to torque data
w=(???)*rpm % convert rpm to rad/s

west=(2*pi/60)*[-3500 3500];
best = ???; % play with this to get the straight line to approximate the data
Test=best*west;

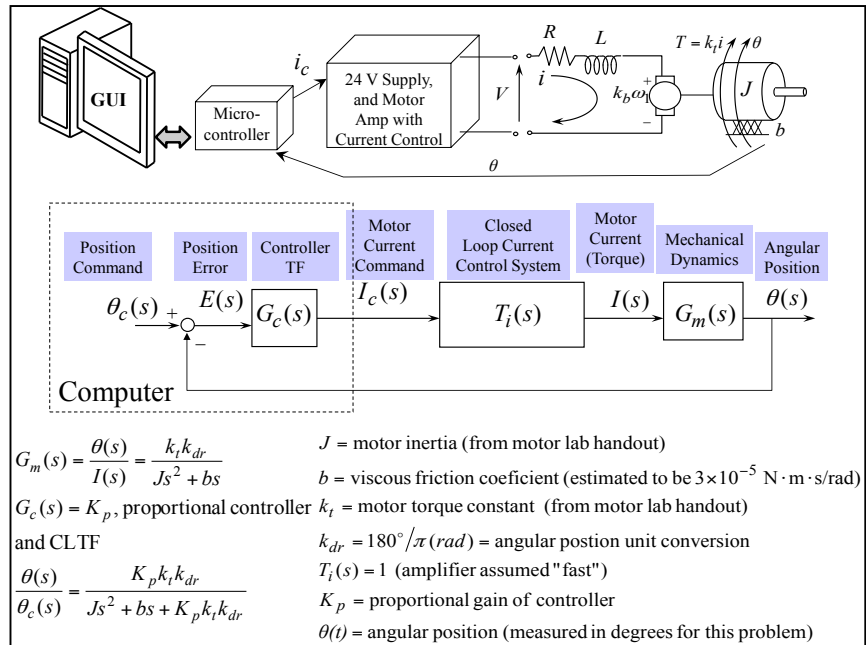
plot(w,T,west,Test);
ylabel('Friction Torque (???)');
xlabel('Angular Velocity (rad/s)');
title('Input Torque vs. Angular Velocity with Estimated Straight Line Fit');

% icresponse part of the file requires "data" to be present in the workspace
% uncomment the lines below to complete the ic response plot

% dataTime=data(:,1); %extract the first column of the data matrix
% dataRPM=data(:,5);
%
% to=???; %time at which torque was shut off in original data
% dataTime=dataTime-to; %shift the time vector of the data to zero at IC
%
% J=1.29e-5;
% tau=J/best; %using your units for J and best is tau in seconds? check it
% Wo=???; %Initial velocity (rpm)
%
% theoryTime=0:0.01:3; %WHAT DOES THIS DO? TRY IT IN THE COMMAND WINDOW. ALSO TYPE HELP COLON
% theoryRPM=Wo*exp(-theoryTime/tau);
%
% figure(2);
% plot(dataTime,???,???,theoryRPM);
% ylabel('Angular Velocity (???)');
% xlabel('Time (sec)');
% title('Actual and Theoretical Response to and IC of ????? rpm');
```

Laboratory #5

In this lab you are to experiment with the position control system of the “Motorlab” apparatus. Also, you are to use a model of the closed-loop position control system to predict the response. You will compare the theoretical step response with the actual response obtained experimentally from the Motorlab. You will compare the responses for three different proportional controller gains. You should also make connections between pole locations and characteristics of the response such as the frequency of oscillation and the decay rate of the oscillations.



Work To Be Done Prior To Lab

- Assuming the transfer function of the closed loop current control system, $T_i(s)$, is one obtain a symbolic representation of the CLTF $\theta(s)/\theta_c(s)$.
- From a) write an equation for the closed-loop poles of the system.
- From b) determine an equation for K_p where the response of the system becomes oscillatory (i.e. where the poles become complex rather than real).
- Plug in the numbers and determine the value of K_p for part c).
- Plug the numbers and the following three gains into your equation for part b) to find the oscillation frequency, and time constant for the decay rate of the oscillations, for each gain. $K_p = 0.01, 0.001, 0.0001(\text{units?})$

Obtaining Data From The Motorlab

In this lab you are using a position control system. Therefore you should run the Motorlab control program in position control mode. For this part of the lab you need to collect experimental data for the step response of the closed-loop system for the three different proportional gains given above. You will have to change the gains and you will have to play with the sample frequency and wave frequency to obtain appropriate data that shows the entire step response. Use the following wave magnitudes for the responses and save the data into the matrix name given. Hint: you should have at least 3 seconds of data on the positive portion of the square wave.

PAY ATTENTION TO THE ORDER!

| Gain, K_p (units?) | Magnitude of Square Wave (degrees) | Name MATLAB workspace matrix for data. |
|----------------------------|------------------------------------------|-------------------------------------------|
| 0.01 | 200 | data3 |
| 0.001 | 1000 | data2 |
| 0.0001 | 10000 | data1 |

Immediately after importing the data to the workspace, plot the data using the “mlposplots(data)” command inside of the MATLAB command window. Check the appropriateness of your sample frequency and wave frequency. Also, use the data cursors to measure the period of oscillation for the table below.

Obtaining the required plots and data

By completing the given m-file code you should generate the required plots for this lab. You should also fill in the table below. Some of the data for this table is generated in the m-file. Other data can be found with the data cursors available in the plots generated with “mlposplots.m”.

| Gain, K_p (units?) | Theoretical CLTF poles, $-\zeta\omega_n \pm j\omega_d$ (rad/s) | Theoretical Period of Oscillations, $2\pi / \omega_d$ (seconds) | Measured Period of Oscillations, T (seconds) | Theoretical Time Constant of Envelope, $1/\zeta\omega_n$ (seconds) | Measured Time Constant of Envelope, τ (estimate one for all three gains) (seconds) |
|----------------------------|-------------------------------------------------------------------------|--------------------------------------------------------------------------|---------------------------------------------------------|-----------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| 0.01 | | | | | |
| 0.001 | | | | | |
| 0.0001 | | | | | |

Things to turn in

- You should have three different plots (with axis labels including units, titles, and legends): 1) simulated unit step response for all three gains, 2) experimental normalized step responses for all three gains, and 3) simulated and experimental response for $K_p=0.01$.
- The completed table.
- Hand development of parts a) thru e).
- The answers to the fill in the blanks below (bold and underlined).

Fill in the blanks (Turn in by end of lab)

1. In the theoretical model, as the proportional gain is increased beyond the value where the closed loop response becomes oscillatory, the damped frequency of oscillation _____ and the time constant for the envelope of the oscillations _____. This captures the behavior of the actual system pretty well, although the envelope does change a little. This might be explained by the nonlinear friction and saturations.
2. As we increase the proportional controller gain beyond 0.001 some aspects of the controller get better while others get much worse. If we try to turn the shaft with our fingers the higher gain system deflects much _____ than the lower gain (try it). This indicates _____ disturbance rejection. However, the damping of oscillations in the step response becomes much _____. This indicates the system is nearly unstable. This is one reason we often add “dynamics” to the controller rather than just the proportional gain which has no integrals or _____.
3. If we keep turning the proportional gain up the system actually becomes _____ (try it). The theoretical model we used doesn’t predict this. There are always more dynamics out there at higher frequency that we haven’t modeled (we’ll look at some in the next lab). For example, by assuming the current controller in the amplifier had a TF of 1, we assumed that it responds _____ fast.
4. Using mlposplots to plot the data in the “data1” matrix we see in the fourth plot, which compares the _____ with the _____ command, that early in the response the current does not actually track the commanded current. As we simulated in the previous lab real systems sometimes have saturations that can affect the response. Looking at the other plots we can see in plot number _____ that the _____ seems to saturate during this period, as can be seen by it reaching a high value and staying constant at that value for a short period. We asked our instructor (do this ☺) and they explained that this is actually due to the limited voltage of the power supply and the _____ constant of the motor. The motor actually generates a voltage as it spins that is proportional to the _____.

Starting m-file code

```
% lab5.m file
% Requires that the square-wave-response data files
% have been imported into data1, data2, and data3.

kt = ???; % N-m/A
J=???; % kg-m^2 or N-m-s^2/rad
b= ???; % N-m-s/rad
kdr=???; % deg/rad

Gm=tf(???);

kp=0.0001;
Gol=kp*Gm;
T1=feedback(Gol,1);
[th1,t1]=step(T1);
[p1,z1]=pzmap(T1)

kp=0.001;
Gol=kp*Gm;
T2=feedback(Gol,1);
[th2,t2]=step(T2);
[p2,z2]=pzmap(T2)

kp=0.01;
Gol=kp*Gm;
T3=feedback(Gol,1);
[th3,t3]=step(T3);
[p3,z3]=pzmap(T3)

dt1=data1(:,1); %extract the time column of the data matrix
dth1=data1(:,3); %extract the first angle column of the data matrix
dth1=dth1/10000; %scale the response to a unit step response

dt2=data2(:,1); %extract the time column of the data matrix
dth2=data2(:,3); %extract the first angle column of the data matrix
dth2=dth2/2000; %scale the response to a unit step response

dt3=data3(:,1); %extract the time column of the data matrix
dth3=data3(:,3); %extract the first angle column of the data matrix
dth3=dth3/200; %scale the response to a unit step response

figure(1); %Theoretical for all three gains
plot(???)

figure(2) %Experimental for all three gains
plot(???)

figure(3) %Experimental and Theoretical for Kp=0.01
plot(???)
```

Laboratory #6

Introduction

In this lab we will use the velocity control system in the Motorlab to look at the concept of "higher frequency dynamics." This lab should illustrate there are always some higher frequency dynamics that will affect you if you "turn up the gains" too much. We can ignore them to a point, but they are there. Often we do not have a good model for them, or even know for sure what is causing them, but they are there.

We have a rule of thumb: *We can ignore open loop poles and zeros when they are more than 10 times larger (in terms of magnitude, which is the distance from the origin of the s plane) than the closed loop poles that result from ignoring them.* You should note it refers to the effect of open loop poles on the closed loop system. This is typical in control system design. We are usually trying make predictions or calculations for the closed loop system using open loop models.

Collecting Data

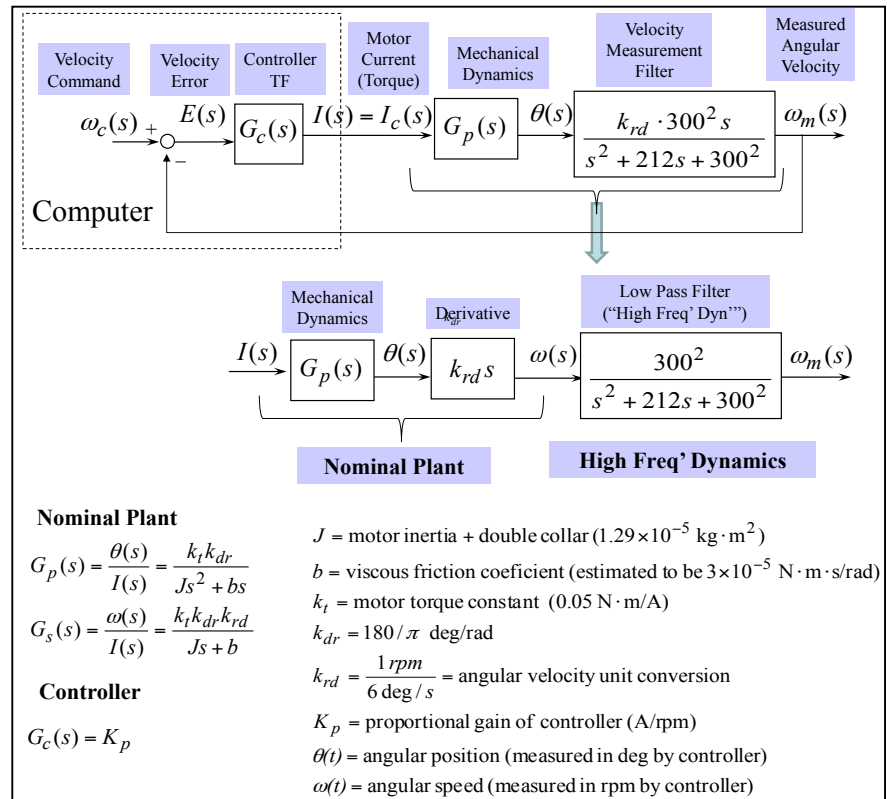
You should collect data for at least three gains, listed in the table below. Use a sample rate of 1000 Hz for all the data. For the first two gains you should collect a step response. For the last gain we want to capture the unstable growth of the response.

| Gain, K_p (units?) | Square Wave (rpm) | Name of MATLAB workspace matrix for data |
|----------------------------|--------------------------------|---------------------------------------------|
| 0.0008 | 1000 | data1 |
| 0.0016 | 1000 | data2 |
| 0.008 | Use Special Procedure below | data3 |

Table 1: Information for Acquiring Data

Special Procedure:

1. Turn off the amplifier.
2. Set the gain to 0.008.
3. Change the rpm to 50 rpm in the manual command window.
4. Turn on the amplifier. Then wait one second and save the data to the workspace.
5. Use mlspeedplots to plot the data and zoom in on the exponential growth in both speed and current plots.



Laboratory #6

Things to Turn In

- Include the two plots the measured and calculated step response when for $K_p = 0.0008$ and $K_p = 0.0016$. These should include separate responses, one measured and the two models of the closed loop system.
- Include a documented copy of your MATLAB code (complete the comments).
- Include the narrative below with the blanks filled in with **bold underlined** answers. Hint: to find the open loop poles of the two systems use “damp (Gs)” and “damp (Gs*???)” in MATLAB.

The “damp” command in MATLAB prints out the poles of a TF in both Cartesian form and polar form. The Cartesian form gives the real and A parts, while the polar form gives the damping ratio and the magnitude. The angle in polar form is directly related to the B.

The nominal system model is the one without the higher frequency dynamics (i.e. without the low pass filter on speed). The nominal C TF has one pole and the nominal closed loop TF has D pole(s). With the higher frequency dynamics the open loop TF has E poles and the closed loop TF has F poles. Closing the loop G change the number of poles. It changes the H of the poles (i.e. they move in the s-plane).

One of the open loop poles in the system with the higher frequency dynamics has the same value as the nominal open loop pole. This pole is much I in magnitude than the other two poles. The magnitude of the other open loop poles in the model with the high frequency dynamics is J rad/s, which is their natural frequency. Therefore according to our rule of thumb, when the K loop poles of the nominal system approach a magnitude of L rad/s we might expect the accuracy of the nominal model to be questionable.

At a gain of $K_p=0.0008$ M the magnitude of the closed loop pole of the nominal model is N rad/s and is close to the pole at O rad/s in the closed loop model with the higher frequency dynamics. At this gain we are very close to our rule of thumb, and although there are slight differences, the step responses from the two models and from the actual system all look very similar. The other two closed loop poles in the model with the filter have complex values of P rad/s, which have a magnitude of Q rad/s. This magnitude is much larger than the magnitude of the real pole and therefore these poles R affect the response much.

At a gain of 0.0016 S the magnitude of the closed loop pole of the nominal model is T rad/s, so it is U than the magnitude predicted by our rule of thumb where we will start to see significant differences between the two models. At this gain the real closed loop pole of the higher order model has a value of -68.8 rad/s, so we might expect similar behavior from the two models. However, the other two closed loop poles in the model with the filter have complex values of V rad/s, which have a magnitude of W rad/s. At this gain the magnitude of the real pole and the complex poles are closer together than for the lower gain and we begin to see the effects of the complex poles with some X in the responses of the higher order model and in the actual system.

At a gain of 0.008 Y the closed loop pole of the nominal model is at Z rad/s, which predicts a fast, first order, stable response. However the higher order model and the actual system are AA, as predicted by the positive real parts of the two closed loop poles at BB rad/s. Although the oscillations in the actual response from the Motorlab do not grow to infinity, they do begin grow and then reach a CC cycle after a few oscillations. It can be seen in plot of the motor current that it saturates at DD Amps.

A. _____
B. _____

C. _____
D. _____
E. _____
F. _____
G. does/does not
H. _____

I. _____
J. _____
K. _____
L. _____

M. (units)
N. _____
O. _____
P. +/- j
Q. _____
R. do/do not

S. (units)
T. _____
U. larger/smaller
V. +/- j
W. _____
X. _____

Y. (units)
Z. _____
AA. _____
BB. +/- j
CC. _____
DD. _____

Laboratory #6

```
% High Frequency Dynamics Lab

kt = 0.05;      % N-m/A
J = 1.29e-5;   % kg-m^2 or N-m-s^2/rad
b = 3e-5;      % N-m-s/rad
kdr = 180/pi;  % deg/rad
krd = 1/6;     % rpm/(deg/s)

Gs = tf([kt*kdr*krd],[J b]);

wn = ;         % Low pass filter in velocity measurement
zwn = ;
Ghf = tf(wn^2,[1 2*zwn wn^2]);

% WHEREEVER YOU SEE ????? IN THE COMMENTS YOU NEED TO COMPLETE
kp=[ 0.0008 0.0016 0.008]; % array of kp gains used
for i=1:length(kp) % cycle through the gains
    Tnominal(i) = feedback(kp(i)*Gs,1); % ????? TF for nominal model
    Thf(i) = feedback(kp(i)*Ghf*Gs,1); % CL TF for higher order model
    display(kp(i)); % display the current kp value
    damp(Tnominal(i)) % show the poles in polar form
    damp(Thf(i)) % ""
    [p1,z1]=pzmap(Tnominal(i)); % CL loop poles and zeros of ????? model
    [p2,z2]=pzmap(Thf(i)); % CL loop poles and zeros of ????? model
    p1= sort(p1); % order the poles small to ?????
    p2 = sort(p2); % ""
    pnominal(:,i) = p1; % add poles for this gain to list
    phf(:,i) = p2; % ""
end

figure(1); % Plot the closed loop poles for each of the gains
hold on;
for i=1:length(kp)
    plot(real(pnominal(:,i)),imag(pnominal(:,i)), '+'); % Nominal model
    plot(real(phf(:,i)),imag(phf(:,i)), 'x'); % Higher order model
end
hold off;
axis equal;
s=sprintf('Poles of nominal CL system, + \n');
s=[s sprintf('and higher order CL system, x \n')];
s=[s sprintf('for three gains \n')];
title(s);
axis equal;

tfinal = .3;
[speedN,timeN]=step(1000*Tnominal(1), tfinal); % step response of nominal model
[speedHF,timeHF]=step(1000*Thf(1), tfinal); % "" of higher order model
speed= data1(:,5); %extract the first speed column of the data matrix
time=data1(:,1); %extract the time column of the data matrix

figure(2);
plot(timeN,speedN,timeHF,speedHF,time,speed); % step response plot kp=0.0008
title('Plot of CL step response for Kp=0.0008');
legend('model with nominal dynamics','model with hi freq dynamics','actual');
xlabel('time (sec)'); ylabel('speed (rpm)');

tfinal = .15;
[speedN,timeN]=step(1000*Tnominal(2), tfinal); % step response of nominal model
[speedHF,timeHF]=step(1000*Thf(2), tfinal); % "" of higher order model
speed= data2(:,5); %extract the first speed column of the data matrix
time=data2(:,1); %extract the time column of the data matrix

figure(3);
plot(timeN,speedN,timeHF,speedHF,time,speed); % step response plot kp=0.0016
title('Plot of CL step response for Kp=0.0016');
legend('model with nominal dynamics','model with hi freq dynamics','actual');
xlabel('time (sec)'); ylabel('speed (rpm)');
```

Laboratory #8

In this lab you are to experiment with the velocity control system of the “Motorlab” apparatus. You are to compare proportional control to PI control, understand the concept of “system type”, and relate the step responses to the poles of the closed loop transfer functions.

“System Type” Background (See pdf)

“System type” for a unity-feedback closed loop system is defined as the number of free integrators in the open loop transfer function. It can be related to steady state errors for different commands (e.g. steps, ramps, parabolas) to the closed loop system.

Velocity Measurement in The Motorlab

To measure velocity in the Motorlab system a filter is used on the position output from the encoder. This filter takes a derivative, and also uses a second order low pass filter to smooth the discrete pulses coming from the encoder, which would cause larger spikes in the derivative. With a cutoff frequency of 300 rad/s, this filter is the higher frequency dynamics that limit the size of the proportional gain.

Obtaining data

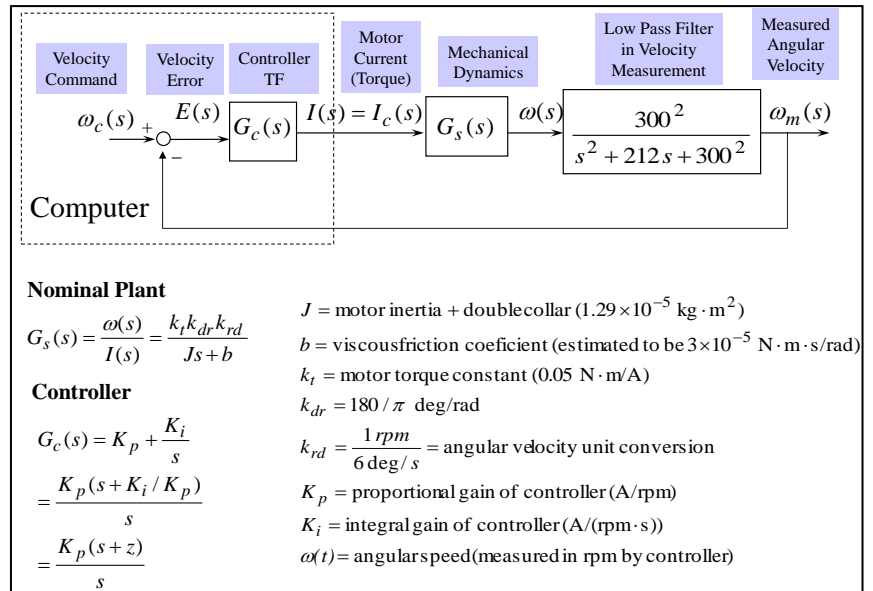
You should obtain the step response from the motor lab for three separate controllers: two proportional controllers and one PI controller.

| Gain, K_p (units?) | Gain, K_i (units?) | Magnitude of Step (rpm) | Matrix name when saved into the MATLAB workspace |
|-------------------------|-------------------------|----------------------------|-----------------------------------------------------|
| 0.0015 | 0 | 1000 | data1 |
| 0.003 | 0 | 1000 | data2 |
| 0.0015 | 0.00405 | 1000 | data3 |

Things to Turn In

- A single plot showing the experimental step responses obtained for the three sets of controller gains.
- The completed table below
- Include the narrative below with the blanks filled in with **bold underlined** answers.

| Controller Gains | | Model | | | | Experimental data |
|-------------------------|-------------------------|--------------------------------|---------------------------------|-----------------------|-----------------------|---------------------------------|
| Gain, K_p (units?) | Gain, K_i (units?) | DC Gain, K_{dc} (rpm/rpm) | Step Response SS Speed (rpm) | CLTF poles (rad/s) | CLTF zeros (rad/s) | Step Response SS Speed (rpm) |
| 0.0015 | 0 | | | | none | |
| 0.003 | 0 | | | | none | |
| 0.0015 | | | | | | |



Narrative

In the table and the plots we find that the experimental responses are very, very similar to the response from the models. So we will use the models for detailed discussion.

With a step input of 1000 RPM, the steady state speeds for the two systems with the proportional controllers are A RPM and B RPM. Using the friction coefficient we can calculate that the input torques required to balance with the friction torque at these two speeds are C N-m and D N-m, respectively. Using the torque constant we can calculate that these two torques correspond to motor currents of E Amps and F Amps. Now, we can look at this from another direction. The output of a proportional controller is the gain multiplied by the error. We find that the steady state outputs of the two controllers should be G (Amps/RPM)*40 (RPM) = H Amps, and I (Amps/RPM)* J (RPM) = K Amps. Therefore, we see that the outputs of the controllers in steady state are balancing with the L torques (currents). And, since a steady state torque (current) is required to maintain speed in this system, there must be a steady state error if we only use M control.

When the integral gain is included, we see that the steady state speed is N RPM, because the DC gain of the CLTF is O. The integral part of the controller continues to grow, by integrating the error, until the steady state error P. With only proportional control we can only decrease the steady state error by Q the gain, which we see causes the system to become more oscillatory and less R. However, we can drive the steady state error to zero with the smaller proportional gain when we include the integral control action, and still maintain a relatively stable closed loop system.

The behavior discussed above can be abstracted to other systems and to other inputs to a closed loop control system. We can use "system type" in this abstraction. We see that the DC gain of the closed loop system is S (i.e. the steady state error for a constant input is T) if the open loop TF has a U. In this lab the open loop transfer function has V free integrators with proportional control, and is therefore type W. It has X free integrator with the PI controller, and is therefore type Y. However, if we put a ramp command into a closed loop system that is type one, it would have a steady state error. A type two system would track a ramp command with zero steady state error. In general we can increase the ability of the closed loop system to track more quickly changing commands by increasing the system type (i.e. by using Z control).

On a different subject, we can relate the transient part (not steady state) of step responses of the three systems to closed loop poles and zeros. The first system is dominated by the real pole at AA rad/s, with the underdamped poles causing BB superimposed on top of the first order response. The real pole gives a time constant of CC seconds, which can be seen in both the step response of the model and the actual system. The second system has a set of complex poles and a real pole, neither of which are DD. The step response looks like a second order response except the first couple of oscillations are not quite symmetric about the steady state value. The first order pole causes them to shade EE. The third system is very similar to the first except it has very near FF at -2.7 rad/s.

A. _____
B. _____
C. _____
D. _____
E. _____
F. _____
G. _____
H. _____
I. _____
J. _____
K. _____
L. _____
M. _____

N. _____
O. _____
P. _____
Q. _____
R. _____

S. _____
T. _____
U. _____
V. _____
W. _____
X. _____
Y. _____
Z. _____

AA. _____
BB. _____
CC. _____
DD. _____
EE. _____
FF. _____

Starting m-file code

```
% lab8 m-file
% Requires that the square-wave-response data files
% have been imported into data1, data2, and data3.

kt = 0.05;      % N-m/A
J=1.29e-5;      % kg-m^2 or N-m-s^2/rad
b= 3e-5;        % N-m-s/rad
kdr=180/pi;     % deg/rad
krd=1/6;        % rpm/(deg/s)

Gs=tf(kt*kdr*krd,[J b]);          % mechanical dyn' with speed output
wn=300; zeta = 0.707/2;
Gvf=tf(wn^2,[1 2*wn*zeta wn^2]); % low pass part of the velocity filter
Gp=Gs*Gvf;

Gc=0.0015;
T1=feedback(Gc*Gp,1);
[p1,z1]=pzmap(T1)
SSspeed = dcgain(T1)*1000

Gc=0.003;
T2=feedback(Gc*Gp,1);
[p2,z2]=pzmap(T2)
SSspeed = dcgain(T2)*1000

Gc = tf(0.0015*[1 2.7],[1 0]);
T3=feedback(Gc*Gp,1);
[p3,z3]=pzmap(T3)
SSspeed = dcgain(T3)*1000

input=data1(:,2);
t1=data1(:,1);
rpm1=data1(:,5); %extract the first speed column of the data matrix

t2=data2(:,1);
rpm2=data2(:,5); %extract the first speed column of the data matrix

t3=data3(:,1);
rpm3=data3(:,5); %extract the first speed column of the data matrix

figure(1) %Experimental for all three gains
plot(t1,input,t1,rpm1,'b',t2,rpm2,'g',t3,rpm3,'r')
legend('StepInput','kp=0.0015','kp=0.003','kp=0.0015,ki=0.0041')
xlabel('Time (s)')
ylabel('Speed Response (rpm)')
title('Step Response of Speed Control with Three Sets of Gains')

[vm1,tm1]=step(1000*T1);
figure(2); plot(t1,rpm1,tm1,vm1)
title('Step responses from actual system and model for kp=0.0015')

[vm2,tm2]=step(1000*T2);
```

```
figure(3); plot(t2,rpm2,tm2,vm2)
title('Step responses from actual system and model for kp=0.003')

[vm3,tm3]=step(1000*T3);
figure(4); plot(t3,rpm3,tm3,vm3)
title('Step responses from actual system and model for PI control')
```

Laboratory #9

In previous labs you experimented with a proportional (P) controller for position control in the Motorlab. We found that as we raised the gain of the P controller that we could obtain somewhat better control of the system, but that the improvement was limited. We could only raise the gain so much before the response became very oscillatory. Furthermore the settling time could not be improved. Now you are to compare the proportional controller to a proportional-derivative (PD) controller. The PD controller adds a zero to the open-loop TF, changing the shape of the root locus. To obtain experimental data you should use the following three controllers.

Three controllers for experimental data

1. P controller, $K_p = 0.001 \text{ Amp/deg}$
2. PD controller, $K_d = 0.00007 \text{ Amp} \cdot \text{s/deg}$, $z = 10 \text{ rad/s}$
3. PD controller, $K_d = 0.001 \text{ Amp} \cdot \text{s/deg}$, $z = 10 \text{ rad/s}$

Obtaining Data From The Motorlab

One problem with real systems (rather than mathematical models) is saturation. When we use a derivative term in the controller, a step input will saturate the output of the controller in a real system. Therefore, we often do not get a good match between experimental results and theoretical models for a step response when derivative control is used. We will discuss this in the lab preparation. **DO NOT LET THIS DISCOURAGE YOU FROM USING DERIVATIVE CONTROL.** It can be very important in obtaining an optimal closed-loop system. Remember that step inputs are usually used as test signals – not as the actual commands in the operation of real systems such as machine tools, aircraft, etc. Because of this problem we will be using a triangle wave for the test command in this lab. **You should obtain the triangle wave response for the three different position control systems. Use a triangle wave with amplitude of 2000 degrees and a wave frequency of 0.5 Hz. You should obtain three different plots.**

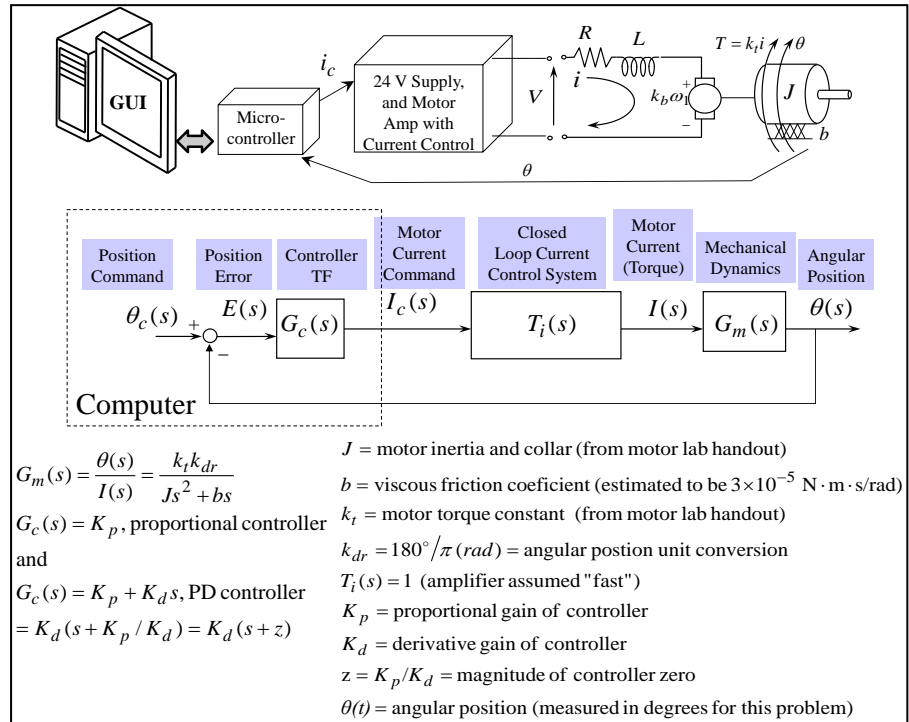
Using MATLAB

Using the “Sisotool” you should play with the systems to get a feel for the responses as the gains change. You should also use the Sisotool to get a feel for root locus. We will do an introduction to the Sisotool in lab. Another related MATLAB function is “rlocus” – try it.

- In the Sisotool, in the MATLAB workspace, or in the m-file, you should find the closed-loop poles and zeros of each of the three systems.
- Outside of the Sisotool, you should also obtain a single step response plot with the response of all three systems.

By hand, using the basic rules, you should draw two root locus plots.

- One plot should be for the P controller.
- The other should be for the PD controller with the zero at -10.
- On the plots clearly indicate the closed loop poles for the three different systems. (i.e. Where on the root locus are you?)



Things to Turn In

- A plot with all three step responses from the models.
- The triangle wave responses (use mposplots or add code to the m-file).
- The root locus plots (by hand).
- Closed-loop poles and zeros for all three systems.
- M-file with comments
- Fill in the blanks (a copy due to instructor before you leave)

Fill in the blanks.

The step responses from the models obviously show that the A set of gains give the fastest response and the triangle wave response from the actual system tracks the best with this set of gains. This can be seen in the B for the three systems. The first system has a pair of complex poles, with a time constant for the envelope of C seconds, and D zeros. The second system has a pair of complex poles with a time constant for the envelope of E seconds, and a zero at F. The third system has G at -10 and a real pole with a time constant of H seconds, which should obviously give the fastest response.

With only proportional control the time constant for the envelope of the resulting complex poles I. We cannot J the settling time with proportional control. By adding a K to the controller with PD control we can pull the poles into the left hand plane.

We can see that the model for the first set of gains can be used to predict aspects of the triangle wave response. The imaginary part of the poles is L rad/s, predicting an oscillation period of M seconds, and we see this in the step response from the model. Also, the period of oscillation from the actual triangle wave response is N seconds (hint: use the two crossings of the response with the command).

With a PD controller and the mechanical plant the system type is O, predicting a P steady state error for a ramp input. Using “mposplots” for the two triangle wave responses with the PD controllers we see in figure 3 that the error settles to a steady state value after each change in direction. This shows up as a Q portion in the error plot. The steady state value for the error with the second set of gains is about R degrees, and for the third set of gains it is about S degrees. We will see in frequency response design that the “open-loop” gain for the third system is much higher, predicting better tracking.

Starting m-file code on the ME 570 website

```
% lab9.m file

kt = 0.05; % N-m/A
J=1.29e-5; % kg-m^2 or N-m-s^2/rad
b= 3e-5; % N-m-s/rad
kdr=180/pi; % deg/rad

Gm=tf(kt*kdr,[J b 0]);

% system with just proportional control
kp = 0.001; Gc1 = kp;
T1=feedback(Gc1*Gm,1);
[th1,t1]=step(T1,1);

kd=0.00007; kp=????; Gc2=tf(????);
T2=feedback(????);
[th2,t2]=step(T2,1);

kd=0.001; kp=????; Gc3=tf(????);
T3=feedback(????);
[th3,t3]=step(T3,1);

plot(t1,th1,t2,th2,t3,th3)

% Related MATLAB tools: rlocus(G), sisotool
```

A. _____
B. _____
C. _____
D. _____
E. _____
F. _____
G. _____ - _____
H. _____

I. _____
J. _____
K. _____

L. _____
M. _____
N. _____

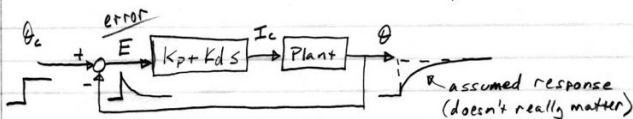
O. _____
P. _____
Q. _____
R. _____
S. _____

Difficulty with Step responses and controller derivatives

The difficulty is that we often don't get a good match between the step response predicted by the model and the step response of the actual system when we use a derivative term in the controller.

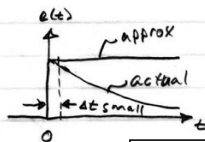
DON'T let this stop you from using derivative control.

Example: Suppose we use a PD controller
 $G_c = K_p + K_d s = K_d (s + z)$, $z = K_p/K_d$



Suppose $\theta_c = 2000 \text{ deg step}$
 $-K_d = 0.0001 \text{ A} \cdot \text{sec/deg}$

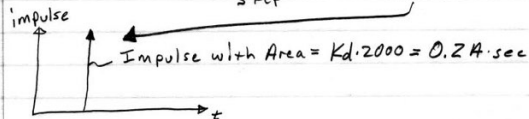
Now approximate $e(t)$ (error) by a step to capture the initial part of $e(t)$



$e(t) = 2000 \text{ deg step} \Rightarrow E(s) = \frac{2000}{s}$

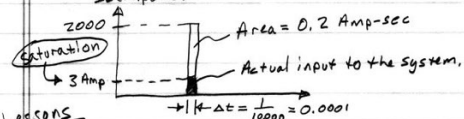
$I_c(s) = E(s) \cdot G_c(s) = \frac{K_p \cdot 2000}{s} + K_d \cdot 2000$

$$\mathcal{L}^{-1}\{I_c(s)\} = \underbrace{K_p \cdot 2000 u(t)}_{\text{step}} + \underbrace{K_d \cdot 2000 \delta(t)}_{\text{impulse}}$$



Computer's Approximation of the Impulse

Computer operates on a clock with 10 KHz frequency, Δt (impulse)



Lessons

- Unless we use very small steps the actual energy that makes it to the system is much smaller than that predicted by the linear model, because of saturation.
- Saturation can also be a factor with large steps and proportional control. It is much easier to calculate when this will happen. $K_p \cdot \text{stepsize} = \text{saturation value}$
- Sometimes we want to use signals other than steps as the test signals.

Example C function for PID control

```
float simple_PID_controller(float error, float delta_time)
{
    float output, Kp=1, Ki=2, Kd=3, max_output=100;
    static float integral, last_error; // static vars for memory

    integral = integral + Ki*error*delta_time; // numerical integration
    if (integral < -max_output) integral = -max_output; // anti-integral windup
    if (integral > max_output) integral = max_output; // anti-integral windup

    output = Kp*error + integral + Kd*(error-last_error)/delta_time; //PID

    last_error = error; // remember for next call

    return(output);
}
```

Laboratory #10

In this lab you are to experimentally determine five data points for the frequency response of the motor-and-spring configuration of the Motorlab. Then you are to estimate all parameters of the model except J and k_{dr} . We assume we know these two parameters accurately.

You are to use sine waves ("Run Wave Autosave ") for the input current on the Motorlab, since the input to the TF of interest is current. **You should use a magnitude of 0.25 Amp for sine wave frequencies near the natural frequency (~resonance).** This will hopefully prevent fatigue failures of the spring. **Be careful near the resonance. It is easy to break the spring with the resonance.** For the other input frequencies you are given and input amplitude to use.

To begin the lab you should experiment to find the actual natural frequency. You should find natural frequency by finding a frequency where the phase lag is very near 90 degrees. If you find a phase lag between 80 and 100 degrees that is sufficient to estimate the natural frequency given that the phase transition is very sharp for this lightly damped system. But try to do your best. Once you have found the natural frequency, then you should fill in the data table. Note that the frequencies you use for data collection are dependent on the natural frequency you find. You may round these other frequencies to the nearest Hz.

Plotting The Responses to Input Sine Waves

You should use the `mlolplots(data, Iscale)` function. You may have to include the `Iscale` argument for the current to be visible on the same plot as the position.

Some Related MATLAB Functions

Helpful MATLAB functions:

`log10()` – log base 10

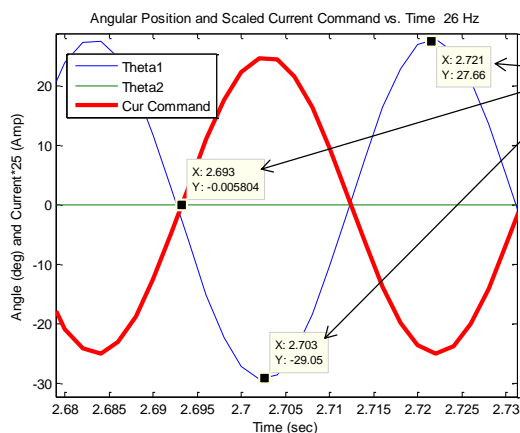
`bode()` – generates the bode (freq' response) plot of a tf – note you can change the freq' units to Hz by right clicking on the figure and choosing 'properties'

`[m,p,w]=bode()` – generates the data for a freq' response plot of a tf – note the mag (m) is a ratio not dB

`loglog()` – plotting routine for a log-log scale

`semilogx()` – plotting routine for a log scale on the x-axis

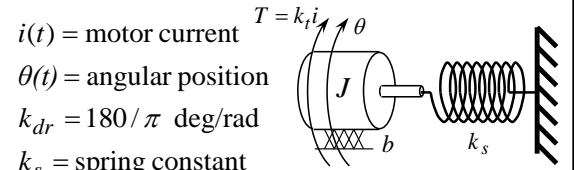
Taking data for the table and searching for the natural frequency.



Use `mlolplots()` to plot the data.

Make three data cursors: one for a zero crossing of the current command, and two the peak and valley of the corresponding crossing of the output. Right click on one of the cursors and choose "Export Cursor Data to Workspace." Then run, for example, `calc_mag_phase(cursors, 26, 2)`.

Here the cursor data was saved to a variable named `cursors`, the input frequency was 26 Hz, and the input amplitude was 2 Amp. Also, in this example the phase lag was about 180 degrees.



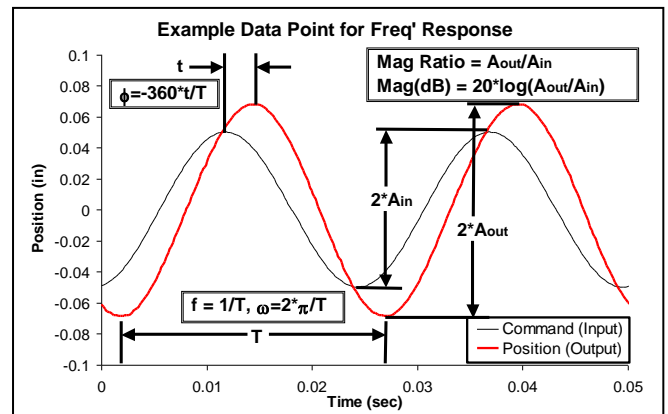
$i(t)$ = motor current
 $\theta(t)$ = angular position
 $k_{dr} = 180/\pi$ deg/rad
 k_s = spring constant
 k_t = motor torque constant
 J = motor inertia + collar inertia
 k_s = spring constant (initial guess
 $\cong 0.385$ N·m/rad)
 b = viscous friction coefficient (initial guess
 $\cong 2 \times 10^{-4}$ N·m·s/rad)

$$G(s) = \frac{\theta(s)}{I(s)} = \frac{k_{dr}k_t}{Js^2 + bs + k_s} = \frac{K_{dc}\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

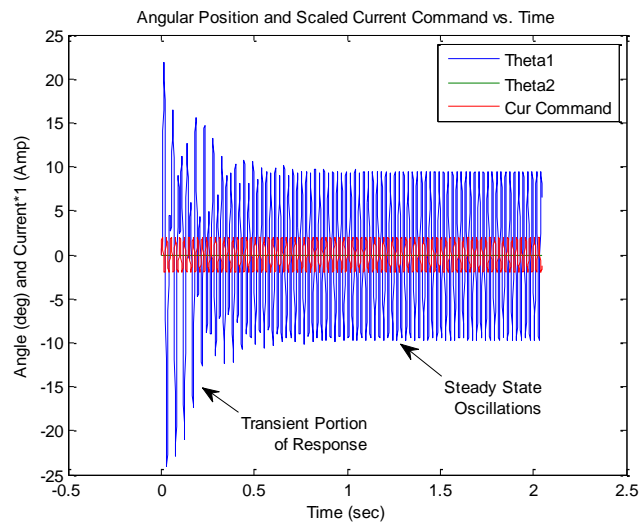
ζ = damping ratio, ω_n = natural frequency

ω_d = damped frequency of oscillation = $\omega_n \sqrt{1 - \zeta^2}$

ω_r = resonant frequency = $\omega_n \sqrt{1 - 2\zeta^2}$



The sample frequency should be chosen so it small enough that there is sufficient time for the response to settle in to steady state oscillations but large enough so that it is at least 10 times larger than the input sine wave frequency. The data should be taken from the steady state oscillations.



DATA TABLE

| Freq' (Hz) | ω_n | $\omega_n / 10$ | $0.75 * \omega_n$ | $1.25 * \omega_n$ | $2 * \omega_n$ |
|-----------------------|------------|-----------------|-------------------|-------------------|----------------|
| Input Amplitude (Amp) | 0.25 | 1 | 1 | 1 | 2 |
| Freq' Value (Hz) | | | | | |
| Mag' Ratio (deg/Amp) | | | | | |
| Mag' Ratio (dB) | | | | | |
| Phase Shift (deg) | | | | | |

Improve your theoretical model

Use data from the table to find all the coefficients for the standard 2nd order form. The magnitude ratio at one tenth of the natural frequency should give you the DC gain. The damping ratio can be found by symbolically calculating the magnitude of the standard 2nd order form at the natural frequency and using then using the actual magnitude at the natural frequency from the data.

Then you should equate the two forms of the model to determine the physical parameters (k_t, k_s, b) of the model.

Things to Turn In

- The Data Table and new estimates for k_t, k_s, b . (a copy turned in to your instructor before you leave).
- Five experimental plots (from `mlolplots()` like on the previous page) of the input and output showing the data cursors used for the "`calc_mag_phase()`" function.
- A final Bode plot showing the initial model, the improved model, and the magnitude and phase data.
- One set of hand-written calculations that duplicate the work done "`calc_mag_phase()`." This should be for the natural frequency and use the data shown in the data cursors.
- A hand development of the magnitude of the standard 2nd order form at the natural frequency.
- Your completed lab 10 m-file.


```

% lab10.m file

% initial model
kt = 0.05;           % N-m/A
kdr = ???;          % deg/rad
J=???;              % kg-m^2 or N-m-s^2/rad
b=???;              % N-m-s/rad
ks = ???;           % N-m/rad

G=tf(????);         % model from initial estimates
figure(1); bode(G)   % generate initial estimate of bode plot
[m,p,w] = bode(G);   % get magnitude, phase, and freq data
m=squeeze(m);        % make m two dimensional
m=20*log10(m);       % convert to dB
p=squeeze(p);

fdata=[????];        % Example - [2 16 21.15 26 42]
wdata=fdata*2*pi;
magdata=[?????];     % Example - [21.9 28.8 48.6 29.1 13.6]
phdata=[?????];      % Example - [-5 -8 -90 -176 -187]

figure(2);
subplot(2,1,1); semilogx(w,m,wdata,magdata,'*')
title('Bode Plot'); ylabel('magnitude (dB)'); xlabel('freq (rad/s)')
subplot(2,1,2); semilogx(w,p,wdata,phdata,'*')
ylabel('phase (deg)'); xlabel('freq (rad/s)')

%%%%%%%%% system id - come up with improved model
wn = ???*2*pi;        % natural freq from data (rad/s)
Kdc = ???;            % dc gain from data
Mwn = ???;            % magnitude ratio at wn from data
zeta = Kdc/Mwn/2;     % calculate damping ratio using Mwn and Kdc
Gnew = ???;

[mnew,pnew,wnew] = bode(Gnew); % get magnitude, phase, and freq data
mnew=squeeze(mnew);
mnew=20*log10(mnew);
pnew=squeeze(pnew);

%plot two models and data together
figure(3);
subplot(2,1,1);
?????

ksnew = ???          % N-m/rad
bnew = ???           % N-m-s/rad
ktnew = ???          % N-m/A

```

```

% calc_mag_phase function file - should not need modification

function calc_mag_phase(cursors,freq,inputmag)

ymin = 1e200; ymax = -1e200; crossing=0;
for i=1:3
    xval = cursors(1,i).Position(1);
    yval = cursors(1,i).Position(2);
    if ((yval > -0.5)&&(yval < 0.5))
        tcrossing = xval;
        crossing = i;
    end
    if (yval > ymax)
        peak = i;
        ymax = yval;
        tpeak = xval;
    end
    if (yval < ymin)
        valley = i;
        ymin = yval;
        tvalley = xval;
    end
end
dtPeakValley = abs(tpeak-tvalley);
outfreq=1/2/dtPeakValley;
if ((crossing==0)|| (valley==peak))
    display('Innacurate cursors or freq!')
    return
end
if ((abs(outfreq-freq)/freq > 0.05)|| (crossing==valley)|| (crossing==peak))
    display('Innacurate cursors or freq!')
    return
end

timelag=mean([tpeak tvalley])-tcrossing;
phaselag = 360*timelag*freq
magratio=(ymax-ymin)/2/inputmag
dB=20*log10(magratio)

end

```

Appendix D

Code

MATLAB

```

%%{
AUTHOR: Derek Black
TOPIC: Noise Characterization
DESCRIPTION: Show that sensor noise frequency is proportional to rotor
speed
%}

%% Extract data
% Time data
time1 = noise1(:,1);
time2 = noise2(:,1);
time3 = noise3(:,1);
time4 = noise4(:,1);
time5 = noise5(:,1);
time6 = noise6(:,1);

% Speed data - Scaled
speed1 = noise1(:,5)/mean(noise1(:,5));
speed2 = noise2(:,5)/mean(noise2(:,5));
speed3 = noise3(:,5)/mean(noise3(:,5));
speed4 = noise4(:,5)/mean(noise4(:,5));
speed5 = noise5(:,5)/mean(noise5(:,5));
speed6 = noise6(:,5)/mean(noise6(:,5));

% Calculated Frequency
f = [4.386 5.55 8.1967 10.8696 13.51 18.5185];
% Speed
s = [mean(noise1(:,5)) mean(noise2(:,5)) mean(noise3(:,5))...
     mean(noise4(:,5)) mean(noise5(:,5)) mean(noise6(:,5))];

% Fit data
x = [0 70];
b = inv(s*s')*s*f';
y = b*x;

%% Comparison figure
figure(1)
subplot(2,1,1)
plot(time1,speed1,'k—')
hold on;
plot(time3,speed3,'k:', 'linewidth',1.2);
hold on;
plot(time6,speed6,'k');
xlim([0 0.5]);
ylim([0.9 1.14]);
xlabel('Time_(Sec)');
ylabel('Normalized_Speed_(rad/s)');
title('Sensor_Noise_Frequency');
legend('12_rad/s','32_rad/s','63_rad/s');
grid on;

subplot(2,1,2)
plot(x,y,'k—',s,f,'x','MarkerSize',5.3);
xlabel('Frequency_(Hz)');
ylabel('Average_Speed_(rad/s)');
legend('Proportionality','Data','Location','southeast');
grid on;

```

Figure D.1: Encoder Noise Experiment

MATLAB

```
%{
AUTHOR: Derek Black
TOPIC: Back EMF estimate
%}

% Extract collected speed data
speedAB = [mean(dataAB1(:,5)) mean(dataAB2(:,5)) mean(dataAB3(:,5)) ...
            mean(dataAB4(:,5)) mean(dataAB5(:,5))]*2*pi/60;
speedBC = [mean(dataBC1(:,5)) mean(dataBC2(:,5)) mean(dataBC3(:,5)) ...
            mean(dataBC4(:,5)) mean(dataBC5(:,5))]*2*pi/60;
speedAC = [mean(dataAC1(:,5)) mean(dataAC2(:,5)) mean(dataAC3(:,5)) ...
            mean(dataAC4(:,5)) mean(dataAC5(:,5))]*2*pi/60;

vab = [3/2 3.44/2 4.2/2 5.32/2 6/2]/sqrt(2);
vbc = [5.2/2 6.44/2 7.48/2 9.48/2 10.1/2]/sqrt(2);
vac = [3.04/2 3.6/2 4.2/2 5.4/2 6/2]/sqrt(2);

% Calculate back emf constants
Kab = inv(speedAB*speedAB')*speedAB*vab';
Kbc = inv(speedBC*speedBC')*speedBC*vbc';
Kac = inv(speedAC*speedAC')*speedAC*vac';
Kavg = (Kab+Kbc+Kac)/3;

speed = mean([speedAB; speedBC; speedAC]);
v = mean([vab; vbc; vac]);

x = [30 speed(5)+3];
y = Kavg*x;

% Plot
figure(1)
plot(speed,v,'x','linewidth',1.5,'MarkerSize',5.5);
hold on;
plot(x,y,'k—');
grid on;
set(gca,'gridlinestyle',':');
xlabel('Motor_Speed_(rad/s)'); xlim([30 75]);
ylabel('Measured_Voltage_(V)');
title('Back_EMF_Constant_Estimate');
legend('Averaged_Data','Ke_Estimate','Location','Southeast');
```

Figure D.2: Back-EMF Plotter

MATLAB

```
%{
AUTHOR: Derek Black
TOPIC: Friction Estimate
DESCRIPTION: Estimate the friction of the NERMLAB in open-loop voltage mode
}%
%% Collected Data
voltage = [-5 -4 -3 -2 -1 -0.5 0 0.5 1 2 3 4 5];
speed    = [mean(sppedn5(:,5)) mean(sppedn4(:,5)) mean(sppedn3(:,5)) ...
            mean(sppedn2(:,5)) mean(sppedn1(:,5)) mean(sppedn0p5(:,5)) ...
            0 mean(spped0p5(:,5)) mean(spped1(:,5)) ...
            mean(spped2(:,5)) mean(spped3(:,5)) mean(spped4(:,5)) ...
            mean(spped5(:,5))];

%% Parameters
kt = 0.01; R = 3.8; Ke = 0.007;
b = 3e-4; J = 5.8e-5;

%% Estimate Friction - Torque
torque = (kt/R)*(voltage-Ke*speed);
estimate = [-50 50];
y = b*estimate;

%% Estimate Friction - Voltage
Kdc = 12.2e-2; % V/(rad/s)
yv = Kdc*estimate;
bv = (kt-(1/Kdc)*Ke*kt)/((1/Kdc)*R); % Convert Kdc to (rad/s)/V

%% Decay
tau = J/b;
w0 = 30; % rad/s
t0 = -1.7;
theoryTime = 0:0.01:3.5;
theoryRAD = w0*exp(-theoryTime/tau);
dataTime = decay(:,1) - t0;
dataRAD = decay(:,5);

%% Plot - Torque vs Speed
figure(1);
subplot(2,1,1); plot(speed,torque,'x','markersize',3.5);
hold on; plot(estimate,y,'k—');
title('Torque_vs_Speed'); ylabel('Torque_(N\cdotm)');
legend('Data','Friction_Estimate','Location','southeast');
grid on; set(gca,'GridLineStyle',':');

%% Plot - Voltage vs Speed
subplot(2,1,2); plot(speed,voltage,'mx','markersize',3.5);
hold on; plot(estimate,yv,'k—');
title('Voltage_vs_Speed'); xlabel('Speed_(rad/s)'); ylabel('Voltage_(V)');
legend('Data','Friction_Estimate','Location','southeast');
ylim([-5 5]); grid on; set(gca,'GridLineStyle',':');

%% Plot - Decay
figure(2); plot(dataTime,dataRAD,'k','linewidth',1.5);
hold on; plot(theoryTime,theoryRAD,'k—');
title('Theoretical/Experimental_Response_to_Initial_Condition');
xlim([-0.15 1]); xlabel('Time_(s)'); ylabel('Speed_(rad/s)');
legend('Data','Theoretical'); grid on; set(gca,'GridLineStyle',':');
```

Figure D.3: Friction Laboratory

MATLAB

```
%{
AUTHOR: Derek Black
DESCRIPTION: Plot root-locus of the open loop system for a position
controller.
%}

%% Constants
kt = 0.034; Ke = 0.04; R = 5;
J = 9.5e-05; b= 4e-4;

%% Models for three seperate gains
Gm=tf(kt,[R*J (R*b+Ke*kt) 0]);
kp=25; Gol=kp*Gm; T1=feedback(Gol,1);
[th1,t1]=step(T1); [p1,z1]=pzmap(T1);

kp= 2.5; Gol=kp*Gm; T2=feedback(Gol,1);
[th2,t2]=step(T2); [p2,z2]=pzmap(T2);

kp=0.9; Gol=kp*Gm;
T3=feedback(Gol,1);
[th3,t3]=step(T3); [p3,z3]=pzmap(T3);

[p,z] = pzmap(Gm);
[r,k] = rlocus(Gm);

realPart = [real(p(1)) real(p(2)) real(p1(1)) real(p1(2)) ...
            real(p2(1)) real(p2(2)) real(p3(1)) real(p3(2))];
imagPart = [imag(p(1)) imag(p(2)) imag(p1(1)) imag(p1(2)) ...
            imag(p2(1)) imag(p2(2)) imag(p3(1)) imag(p3(2))];

xaxis = [-8 1]; xaxisy = [0 0];
yaxis = [-50 50]; yaxisx = [0 0];
names = [ 'Open-Loop-Poles ', 'Kp=0.9 ', 'Kp=2.5 ', 'Kp=25 ']; h = [];
figure(1);
lineStyles = distinguishable_colors(50,'w');
ylim([-50 50])
xlim([-8 1])
for i = 3:2:(length(realPart))
    linestyle = lineStyles(i+38,:);
    hold on
    h(i) = [plot(realPart(i),imagPart(i),'x','Color',linestyle,'markers',9)];
    hold on
    plot(realPart(i+1),imagPart(i+1),'x','Color',linestyle,'markers',9);
end
hold on; h(1) = [plot(realPart(1),imagPart(1),'x','Color','red','markers',9)];
hold on; plot(realPart(2),imagPart(2),'x','Color','red','markers',9);
hold on
plot(xaxis,xaxisy,'k',yaxisx,yaxis,'k','linewidth',0.5);
hold on;
plot(real(r),imag(r),'Color',lineStyles(24,:), 'linewidth',1);
legend([h(1) h(3) h(5) h(7)],{ 'Open-Loop-Poles ', 'Kp=25 ', ...
    'Kp=2.5 ', 'Kp=0.9 '}, 'Location', 'southwest');
hold on;
title('Root-Locus'); xlabel('Real-Axis-(seconds^{-1})');
ylabel('Imaginary-Axis-(seconds^{-1})');
set(gca,'ytick',[-50 -40 -30 -20 -10 0 10 20 30 40 50]);
grid on; set(gca,'gridlinestyle',':'); box on;
```

Figure D.4: Root Locus Plotter

MATLAB

```
%{
AUTHOR: Derek Black
TOPIC: System Characteristics of Position Control System
DESCRIPTION: Increase Kp to observe the behavior of the system.
%}
%% Constants
Ke = 0.007; kt = 0.01; R = 3.8;
J = 5.8e-5; b = 3e-4;

%% Models for three seperate gains
Gm=tf(kt,[R*J (R*b+Ke*kt) 0]);
kp = 25; Gol=kp*Gm; T1=feedback(Gol,1);
[th1,t1]=step(T1); [p1,z1]=pzmap(T1);

kp = 2.5; Gol=kp*Gm; T2=feedback(Gol,1);
[th2,t2]=step(T2); [p2,z2]=pzmap(T2);

kp = 0.9; Gol=kp*Gm;
T3=feedback(Gol,1);
[th3,t3]=step(T3); [p3,z3]=pzmap(T3);

tau = 0.365; Ge=tf(1/tau,[1 1/tau]); [the,te] = step(Ge); % Theory
Ge = -tf(1/tau,[1 1/tau]); [invthe,invte] = step(Ge+2); % Inv Theory
tau = 0.365; Ge=tf(1/tau,[1 1/tau]); [thee,tee] = step(Ge); % Experiment
Ge = -tf(1/tau,[1 1/tau]); [invthee,invtee] = step(Ge+2); % Inv Exp.
%% Extract Data
dt1 = data1(:,1); dth1 = data1(:,3); dth1 = dth1/0.27;
dt2 = data2(:,1); dth2 = data2(:,3); dth2 = dth2/3.22;
dt3 = data3(:,1); dth3 = data3(:,3); dth3 = dth3/8.665;

%% Plot figures
lineStyles = distinguishable_colors(20);
figure(1);
plot(t1,th1,'k');
hold on; plot(t2,th2,'k-.','linewidth',2);
hold on; plot(t3,th3,'k','linewidth',2);
hold on; plot(te,the,'k:',invte,invthe,'k:');
legend('Kp=25','Kp=2.5','Kp=0.9','Envelope');
xlabel('Time_(sec)'); ylabel('\theta_(normalized)');
title('Theoretical_Step_Responses'); xlim([0 1.3]);

figure(2);
plot(dt1,dth1,'k');
hold on; plot(dt2,dth2,'k-.','linewidth',2);
hold on; plot(dt3,dth3,'k','linewidth',2);
hold on; plot(tee,thee,'k:',invtee,invthee,'k:');
legend('Kp=25','Kp=2.5','Kp=0.9','Envelope'); xlabel('Time_(sec)');
ylabel('\theta_(normalized)');
title('Experimental_Step_Responses');
xlim([0 1.3]); ylim([0 2]);

figure(3); plot(dt2,dth2,'k',t2,th2,'k—','linewidth',0.8);
legend('Experimental','Model'); xlabel('Time_(sec)'); ylabel('\theta_(normalized)');
title('Experimental_vs_Theoretical_Step_Response_(Kp=2.5)');
xlim([0 1.3]); grid on; set(gca,'gridlinestyle',':');

figure(4); plot(dt1,dth1,'k',t1,th1,'k—','linewidth',0.8);
legend('Experimental','Model'); xlabel('time_(sec)'); ylabel('\theta_(normalized)');
title('Experimental_vs_Theoretical_Step_Response_(Kp=25)');
xlim([0 1.3]); grid on; set(gca,'gridlinestyle',':');
```

Figure D.5: Frequency of Oscillation Position Control Code

MATLAB

```
%{
AUTHOR: Derek Black
TOPIC: Frequency Response of a Position Controller on NERMLAB
DESCRIPTION: Characterize the NERMLAB system by generating a frequency
response.
}%

%% Constants
Ke = 0.007; kt = 0.01; R = 3.8;
J = 5.8e-5; b = 3e-4; Kp = 20;

Gm=tf(kt,[R*J (R*b+Ke*kt) 0]); T = feedback(Kp*Gm,1);
figure(1); bode(T)
[m,p,w] = bode(T);
m=squeeze(m); m=20*log10(m); p=squeeze(p);

f_exp1 = [0.42 3.43 4.4 5.73 9.0]*2*pi; f_exp2 = [0.44 3.45 4.6 5.75 9.2]*2*pi;
f_exp3 = [0.48 3.47 4.8 5.78 9.4]*2*pi; f_exp4 = [0.44 3.45 4.6 5.75 9.2]*2*pi;
m_exp1 = [-0.74 3.7 6.5 4 -5.3]; m_exp2 = [-0.56 3.8 7.88 2.32 -6.5];
m_exp3 = [-0.69 3.8 11.2 4.5 -7]; m_exp4 = [-0.65 3.7 10 5.1 -4.96];
p_exp1 = [3.3 -18.5 -66.3 -141.1 -171.8]; p_exp2 = [3.3 -31.2 -80.6 -142.4 -176.6];
p_exp3 = [3.4 -32 -99.4 -141.1 -176.6]; p_exp4 = [0 -28.6 -90 -141.5 -170.68];
f_avg = []; m_avg = []; p_avg = [];

for i=1:length(f_exp1)
    f_avg(i) = [mean([f_exp1(i) f_exp2(i) f_exp3(i) f_exp4(i)])];
    m_avg(i) = [mean([m_exp1(i) m_exp2(i) m_exp3(i) m_exp4(i)])];
    p_avg(i) = [mean([p_exp1(i) p_exp2(i) p_exp3(i) p_exp4(i)])];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% system id - come up with improved model
wn = 4.6*2*pi; % natural freq from data (rad/s)
Kdc = db2mag(m_avg(1)); % dc gain from data
Mwn = db2mag(m_avg(3)); % magnitude ratio at wn from data
zeta = Kdc/Mwn/2; % calculate damping ratio using Mwn and Kdc
Gnew = tf(Kdc*wn^2,[1 2*zeta*wn wn^2]);

[mnew,pnew,wnew] = bode(Gnew); % get magnitude, phase, and freq data
mnew=squeeze(mnew); mnew=20*log10(mnew); pnew=squeeze(pnew);

list = get(gca,'colororder');
figure(1);
set(gcf,'DefaultAxesColorOrder','remove');
subplot(2,1,1);
semilogx(w,m,'—','linewidth',2); hold on;
semilogx(wnew,mnew,'linewidth',2); hold on;
semilogx(f_exp1,m_exp1,'x','markers',9); hold on;
semilogx(f_exp2,m_exp2,'*','markers',9); hold on;
semilogx(f_exp3,m_exp3,'o','markers',9); hold on;
semilogx(f_exp4,m_exp4,'+','markers',9); hold on;
semilogx(f_avg, m_avg,'s','markers',9); hold on;
ylabel('magnitude_(dB)'); grid on; title('Bode_Plot');
legend('Model','Improved_Model','Experiment_1','Experiment_2','Experiment_3','Experiment_4',...
'Average','Averaged_Model');

subplot(2,1,2);
semilogx(w,p,'—','linewidth',1,'Color',list(1,:), 'linewidth',2); hold on;
semilogx(wnew,pnew,'Color',list(2,:), 'linewidth',2); hold on;
semilogx(f_exp1,p_exp1,'x','Color',list(3,:), 'markers',9); hold on;
semilogx(f_exp2,p_exp2,'*','Color',list(4,:), 'markers',9); hold on;
semilogx(f_exp3,p_exp3,'o','Color',list(5,:), 'markers',9); hold on;
semilogx(f_exp4,p_exp4,'+','Color',list(6,:), 'markers',9); hold on;
semilogx(f_avg, p_avg, 's','Color',list(7,:), 'markers',9);
ylabel('phase_(deg)'); xlabel('freq_(rad/s)'); grid on;
```

Figure D.6: Frequency Response of a Position Control System Code

MATLAB

```
%{
AUTHOR: Derek Black
TOPIC: Steady State Error
%}

Ke = 0.007; kt = 0.01; R = 3.8; % Electrical
J = 5.8e-5; b = 3e-4; % Mechanical
G = tf(kt,[R*(J*(b+Ke*kt))]);
wn=300; zeta = 0.707/2; Gvf=tf(wn^2,[1 2*wn*zeta wn^2]);
Gp=G*Gvf;

stepSize = 50; stepSizeFinal = 2.6;
Gc = [0.1 0.15 3]; Ki = 1;

T1=feedback(Gc(1)*Gp,1); [p1,z1]=pzmap(T1)
SSspeed = dcgain(T1)*stepSize; [vm1,tm1]=step(stepSize*T1);

T2=feedback(Gc(2)*Gp,1); [p2,z2]=pzmap(T2);
SSspeed = dcgain(T2)*stepSize; [vm2,tm2]=step(stepSize*T2);

T4=feedback(Gc(3)*Gp,1); [p4,z4]=pzmap(T4);
SSspeed = dcgain(T4)*stepSizeFinal; [vm4,tm4]=step(stepSizeFinal*T4);

Gc = tf(Gc(1)*[1 Ki/Gc(1)],[1 0]);
T3=feedback(Gc*Gp,1); [p3,z3]=pzmap(T3);
SSspeed = dcgain(T3)*stepSize; [vm3,tm3]=step(stepSize*T3);

% Extract Data
t1 = data1(:,1); t2 = data2(:,1); t3 = data3(:,1); t4 = data6(:,1);
s1 = data1(:,5); s2 = data2(:,5); s3 = data3(:,5); s4 = data6(:,5);
input1 = data1(:,2); input2 = data6(:,2);

figure(1); plot(t1,input1,t1,s1,'k:',t2,s2,'k—',t3,s3,'k')
title('Experimental_Step_Responses')
legend('Step_Input','Kp=0.1','Kp=0.15','PI','location','southeast');
xlabel('Time(sec)'); ylabel('Amplitude(rad/s)'); xlim([0 0.8]);

figure(2); plot(t1,s1,'k',tm1,vm1,'k—')
title('Actual_System_and_Model_for_Kp=0.1')
legend('Actual','Model','location','southeast')
xlabel('Time(s)'); xlim([0 0.6]); ylabel('Velocity(RPM)');

figure(3); plot(t2,s2,'k',tm2,vm2,'k—')
title('Actual_System_and_Model_for_Kp=0.15')
legend('Actual','Model','location','southeast')
xlabel('Time(s)'); ylabel('Velocity(RPM)'); xlim([0 0.5]);

figure(4); plot(t3,s3,'k',tm3,vm3,'k—')
title('Actual_System_and_Model_for_PI_control')
legend('Actual','Model','location','southeast')
xlabel('Time(s)'); xlim([0 0.8]); ylabel('Velocity(RPM)');

figure(5); plot(t4,input2,t4,s4,'k',tm4,vm4,'k—');
title('Actual_System_and_Model_for_Kp=3')
legend('Step_Input','Actual','Model','location','southeast')
xlabel('Time(s)'); ylabel('Velocity(RPM)'); xlim([0 0.17]);
```

Figure D.7: Steady State Error Code

MATLAB

```
%{
AUTHOR: Derek Black
TOPIC: Proportional-Derivative Control
%}

%% Constants
Ke = 0.007; kt = 0.01; R = 3.8;
J = 5.8e-5; b = 3e-4; z = 25;

%% Model
Gm=tf(kt,[R*J (R*b+Ke*kt) 0]);

kp = 7; Gc1 = kp;
T1=feedback(Gc1*Gm,1);
[th1,t1]=step(T1,1);
[p1,z1] = pzmap(T1)

kd = 0.2; kp = 3
Gc2=tf([kd kp],1);
T2=feedback(Gc2*Gm,1);
[th2,t2]=step(T2,1);
[p2,z2] = pzmap(T2)

kd = 0.6; kp = 9;
Gc3=tf([kd kp],1);
T3=feedback(Gc3*Gm,1);
[th3,t3]=step(T3,1);
[p3,z3] = pzmap(T3)

figure(1); plot(t1,th1,'k',t2,th2,'k—',t3,th3,'k:');
title('Step Response of the Three Models')
xlabel('Time(sec)'), ylabel('Amplitude(rad)')
legend('Kp=7','Kd=0.2,z=15','Kd=0.8,z=15')

% Extract data
t1 = data1(:,1); t2 = data4(:,1); t3 = data3(:,1);
y1 = data1(:,3); y2 = data4(:,3); y3 = data3(:,3);
i1 = data1(:,2); i2 = data4(:,2); i3 = data3(:,2);

figure(2); plot(t1,i1,'k—',t1,y1,'k');
xlabel('Time(sec)'); ylabel('Amplitude(rad)'); title('Kp=7');
legend('Input','Actual'); xlim([0 3.2]); ylim([-3.2 3.2]);

figure(3); plot(t2,i2,'k—',t2,y2,'k');
xlabel('Time(sec)'); ylabel('Amplitude(rad)'); title('Kp=3,Kd=0.2');
legend('Input','Actual'); xlim([0 3.2]); ylim([-3.2 3.2]);

figure(4); plot(t3,i3,'k—',t3,y3,'k');
xlabel('Time(sec)'); ylabel('Amplitude(rad)'); title('Kp=9,Kd=0.8');
legend('Input','Actual'); xlim([0 3.2]); ylim([-3.2 3.2]);
```

Figure D.8: Proportional-Derivative Experiment Code

Appendix E

System Specifications

Table E.1: NERMLAB Parameters

| Parameter | Description | Value | Units |
|-----------|-------------------------------------------|-----------------------|---------------------------------|
| J | Lumped Inertia ($3J_w + J_r + 4J_b$) | 5.8×10^{-5} | $kg \cdot m^2$ |
| J_w | Washer Inertia | 1.7×10^{-5} | $kg \cdot m^2$ |
| J_r | Rotor Inertia | 4.5×10^{-6} | $kg \cdot m^2$ |
| b | Viscous Friction | 3.0×10^{-4} | $\frac{N \cdot m \cdot s}{rad}$ |
| k_t | Motor Torque Constant | 0.01 | $\frac{N \cdot m}{A}$ |
| L | Motor Inductance | 7.58×10^{-4} | H |
| R | Motor Phase Resistance | 3.8 | Ω |
| K_E | Back-EMF Constant | 0.007 | $\frac{V \cdot s}{rad}$ |

Table E.2: Motorlab Parameters

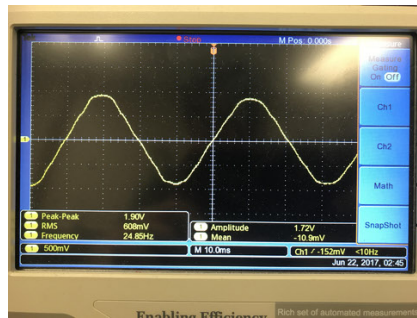
| Parameter | Description | Value | Units |
|-----------|------------------------|--------------------|---------------------------------|
| Inertia | Single Shaft Collar | 15 | $g \cdot cm^2$ |
| Inertia | Double Shaft Collar | 19 | $g \cdot cm^2$ |
| Inertia | Rotor | 110 | $g \cdot cm^2$ |
| b | Viscous Friction | 3×10^{-5} | $\frac{N \cdot m \cdot s}{rad}$ |
| k_T | Motor Torque Constant | 5 | $\frac{N \cdot cm}{A}$ |
| L | Motor Inductance | 4.4 | mH |
| R | Motor Phase Resistance | 1.18 | Ω |
| K_e | Back-EMF Constant | 5.2 | $\frac{V}{krpm}$ |

Appendix F

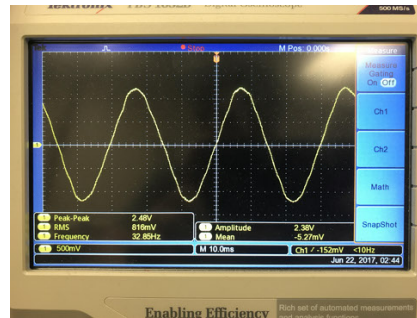
Experiment Documentation

Table F.1: Viscous Friction Full Experimental Results

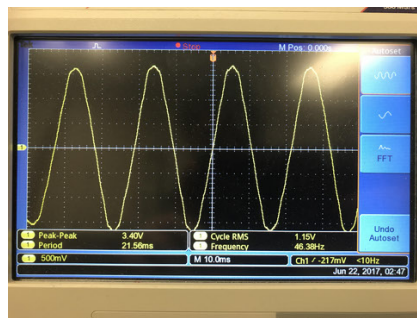
| Voltage (V) | Speed (rad/s) |
|-------------|---------------|
| -5 | -45.0 |
| -4 | -35.0 |
| -3 | -25.2 |
| -2 | -15.5 |
| -1 | -5.8 |
| -0.5 | 0 |
| 0 | 0 |
| 0.5 | 0 |
| 1 | 5.5 |
| 2 | 14.5 |
| 3 | 23.2 |
| 4 | 31.7 |
| 5 | 40.0 |



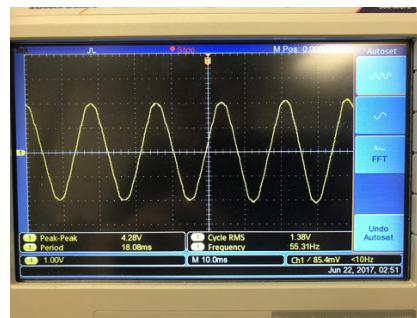
(a) 200 RPM



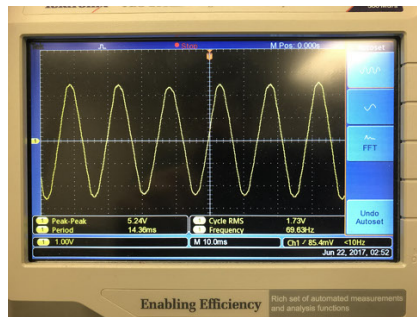
(b) 300 RPM



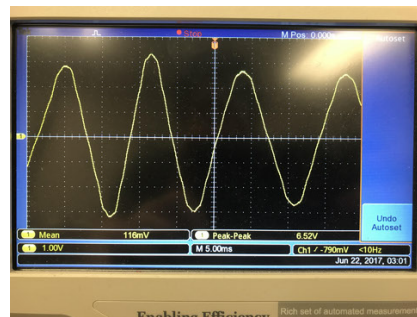
(c) 400 RPM



(d) 500 RPM



(e) 600 RPM



(f) 850 RPM

Figure F.1: Back emf at various motor speeds

Appendix G

Additional Models

Appendix G hosts additional models that were not used in the making of this thesis. They serve as models for either future experiments, or for readers wishing to conduct additional experiments.

G.1 Pendulum Model

The pendulum model is a mathematical formulation for NERMLAB that is used to help find the resonant frequency. It should be noted that the dynamic equations for this setup are non-linear in nature, and as a result, the small angle approximation ($\sin(\theta) \approx \theta$) must be used in order to transform the time dependent system into the frequency domain. The small angle approximation is only valid for angles that are smaller than 0.244 radians, as the relative error between $\sin(\theta)$ and θ exceeds 1 percent at this value. This will be taken into account when running experiments on the NERMLAB system.

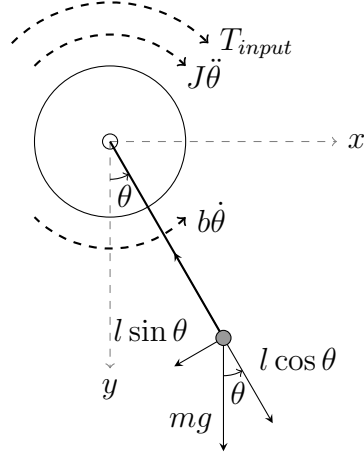


Figure G.1: NERMLAB Pendulum Model

The best way to start the derivation is to find the describing differential equations for both the mechanical and electrical system. In the case of the pendulum setup, the electrical dynamics are simply the torque the motor provides the pendulum with, which was found in section 4.2, equation 4.6.

$$J\ddot{\theta}(t) = -mgl\theta(t) - b\dot{\theta}(t) + T_{input} \quad (\text{G.1})$$

Substituting the electrical motor torque (eq. 4.6), into equation G.1.

$$J\ddot{\theta}(t) = -mgl\theta(t) - b\dot{\theta}(t) + \frac{k_T}{R}(V(t) - K_e\dot{\theta}(t)) \quad (\text{G.2})$$

$$J\ddot{\theta}(t) + mgl\theta(t) + b\dot{\theta}(t) + \frac{k_T K_e}{R}\dot{\theta}(t) = \frac{k_T}{R}V(t)$$

Taking the Laplace transform of the above equation yields:

$$(Js^2 + mgl + bs + \frac{k_T K_e}{R}s)\theta(s) = \frac{k_T}{R}V(s)$$

$$\frac{JRs^2 + mglR + bRs + k_T K_e s}{R} \theta(s) = \frac{k_T}{R} V(s)$$

$$\frac{\theta(s)}{V(s)} = \frac{k_T}{JR s^2 + (bR + k_T K_e)s + mglR} \quad (\text{G.3})$$

Putting equation G.3 into a standard second order form yields:

$$\frac{\theta(s)}{V(s)} = \frac{\frac{k_T}{JR}}{s^2 + (\frac{b}{J} + \frac{k_T K_e}{JR})s + \frac{mgl}{J}} \quad (\text{G.4})$$

To further simplify the model, lumped coefficients for friction and the trailing term of equation G.4 will be made since they are comprised of statically defined variables.

$$b_l = \frac{b}{J} + \frac{k_T K_e}{JR} \quad (\text{G.5})$$

$$\omega_p = \frac{mgl}{J} \quad (\text{G.6})$$

$$k_{T,l} = \frac{k_T}{JR} \quad (\text{G.7})$$

$$\frac{\theta(s)}{V(s)} = \frac{k_{T,l}}{s^2 + b_l s + \omega_p} \quad (\text{G.8})$$

It should be noted that J is the pendulum's inertia and not the rotors in equation G.8; likewise, l is the distance to the center of mass of the pendulum.

G.2 State Space Models

Section G.2 will develop state space models for the open loop position and speed models. Additionally, the closed loop state space models for the position and speed control will be

developed as well. Here, u will represent the input to the system, and in the case of the NERMLAB, is voltage $V(t)$. State space models can be written compactly with formulas G.9, where $\dot{\mathbf{q}}$ and y are the state equation and output equation, respectively. In all cases in this thesis, D is assumed to be zero.

$$\begin{cases} \dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{B}u \\ y = \mathbf{C}\mathbf{q} + Du \end{cases} \quad (\text{G.9})$$

G.2.1 Position Models

Open Loop

To start the derivation of the state space models for open loop position, it is easiest to start with the time domain differential equation, Equation G.10, which was described in Chapter 4.

$$J\ddot{\theta}(t) + \left(\frac{k_t K_E}{R} + b\right)\dot{\theta}(t) = \frac{k_t}{R}V(t) \quad (\text{G.10})$$

From here it is possible to define the states of the differential equation (Equation G.10):

$$\begin{cases} q_1 = \theta \\ q_2 = \dot{q}_1 = \dot{\theta} \\ q_3 = \dot{q}_2 = \ddot{\theta} \end{cases} \quad (\text{G.11})$$

Rearranging Equation G.10, substituting in States G.11 and solving for q_3 :

$$q_3 = \dot{q}_2 = \ddot{\theta} = -\left(\frac{k_t K_E}{RJ} + \frac{b}{J}\right)q_2 + \frac{k_t}{RJ}u \quad (\text{G.12})$$

It is now possible to rewrite the starting differential equation as two first order equations:

$$\left\{ \begin{array}{l} \dot{q}_1 = q_2 = \dot{\theta} \\ \dot{q}_2 = q_3 = \ddot{\theta} = -(\frac{k_t K_E}{RJ} + \frac{b}{J})q_2 + \frac{k_t}{RJ}u \end{array} \right. \quad (\text{G.13})$$

Using the equation forms G.9, it is possible to write the following state space representation of an open loop position system.

$$\left\{ \begin{array}{l} \mathbf{A} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{k_t K_E}{RJ} - \frac{b}{J} \end{bmatrix} \\ \mathbf{B} = \begin{bmatrix} 0 \\ \frac{k_t}{RJ} \end{bmatrix} \\ \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \mathbf{D} = 0 \end{array} \right. \quad (\text{G.14})$$

Closed Loop

For the closed loop position control model, the only variable that changes is \mathbf{A} , which in the closed loop system becomes \mathbf{A}_{cl} by the relationship:

$$\mathbf{A}_{cl} = \mathbf{A} - \mathbf{BK} \quad (\text{G.15})$$

Here \mathbf{A} and \mathbf{B} are still defined as they were in Equations G.14. \mathbf{K} is the gain matrix:

$$\mathbf{K} = \begin{bmatrix} K_1 & K_2 \end{bmatrix} \quad (\text{G.16})$$

Therefore, with Equation G.15, it is possible to define the closed loop position control

state space representation as:

$$\left\{ \begin{array}{l} \mathbf{A}_{cl} = \begin{bmatrix} 0 & 1 \\ -\frac{K_1 k_t}{RJ} & -(\frac{k_t K_E}{RJ} + \frac{b}{J} + \frac{K_2 k_t}{RJ}) \end{bmatrix} \\ \mathbf{B} = \begin{bmatrix} 0 \\ \frac{k_t}{RJ} \end{bmatrix} \\ \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \mathbf{D} = 0 \end{array} \right. \quad (\text{G.17})$$

A unique results appears when setting $\mathbf{K} = \begin{bmatrix} 1 & 0 \end{bmatrix}$, which causes the system to have unity feedback.

$$\left\{ \begin{array}{l} \mathbf{A}_{cl} = \begin{bmatrix} 0 & 1 \\ -\frac{K_1 k_t}{RJ} & -(\frac{k_t K_E}{RJ} + \frac{b}{J}) \end{bmatrix} \\ \mathbf{B} = \begin{bmatrix} 0 \\ \frac{k_t}{RJ} \end{bmatrix} \\ \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \mathbf{D} = 0 \end{array} \right. \quad (\text{G.18})$$

G.2.2 Speed Models

Open Loop

The derivation for the open loop speed model is nearly identical to the one developed in the previous section. Only the describing differential equation changes, which results in equation G.19.

$$J\dot{\omega}(t) + \left(\frac{k_t K_E}{R} + b\right)\omega(t) = \frac{k_t}{R}V(t) \quad (\text{G.19})$$

Here the states of the system can be written as:

$$\left\{ \begin{array}{l} q_1 = \omega \\ q_2 = \dot{q}_1 = \dot{\omega} \end{array} \right. \quad (\text{G.20})$$

Since it only requires one state to describe the state space, since $\dot{q}_1 = \dot{\omega}$, the result is rather trivial and the representation will be directly described below in Equations G.21.

$$\left\{ \begin{array}{l} \mathbf{A} = \left[-\left(\frac{k_t K_E}{RJ} + \frac{b}{J}\right) \right] \\ \mathbf{B} = \left[\frac{k_t}{RJ} \right] \\ \mathbf{C} = \left[1 \right] \\ \mathbf{D} = 0 \end{array} \right. \quad (\text{G.21})$$

Closed Loop

$$\left\{ \begin{array}{l} \mathbf{A}_{cl} = \left[-\left(\frac{k_t K_E}{R J} + \frac{b}{J} + \frac{K k_t}{R J} \right) \right] \\ \mathbf{B} = \left[\frac{k_t}{R J} \right] \\ \mathbf{C} = \left[1 \right] \\ \mathbf{D} = 0 \end{array} \right. \quad (\text{G.22})$$